A theory of processes with localities

Gérard Boudol, Ilaria Castellani, Matthew Hennessy, Astrid Kiehn

UNITÉ DE RECHERCHE
INRIA-SOPHIA ANTIPOLIS

Rapports de Recherche

1992

25ème anniversaire

N° 1632

Programme 2
Calcul Symbolique, Programmation
et Génie logiciel

A THEORY OF PROCESSES WITH LOCALITIES

Gérard BOUDOL
Ilaria CASTELLANI
Matthew HENNESSY
Astrid KIEHN

Mars 1992

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

A Theory of Processes with Localities*

G. Boudol, I. Castellani
INRIA, Sophia-Antipolis,
M. Hennessy
CSAI, University of Sussex,
A. Kiehn
TUM, Munich.

Abstract

We study a notion of observation for concurrent processes which allows the observer to see the distributed nature of processes, giving explicit names for the location of actions. A general notion of bisimulation related to this observation of distributed systems is introduced. Our main result is that these bisimulation relations, particularized to a process algebra extending CCS, are completely axiomatizable. We discuss in details two instances of location bisimulations, namely the location equivalence and the location preorder.

1 Introduction

A distributed system may be described as a collection of computational activities spread among different sites or *localities*, which may be physical or logical. Such activities are viewed as being essentially independent from each other, although they may require to synchronize or communicate at times. It has been argued in previous work [CH89,Cas88,Kie89,BCHK91] that the standard interleaving approach to the semantics of concurrent systems may not be adequate to model such distributed computations: more precisely, it may not be able to express naturally properties of distributed systems which depend on their distribution in space, like e.g. a local deadlock, that is a deadlock in a specific site of the system.

Most noninterleaving semantics proposed so far in the literature for algebraic languages such as CCS [Mil80,Mil89] are based on the notion of *causality* between actions, or on the complementary notion of causal independence or concurrency. Here we pursue the different approach of [CH89,Cas88,Kie89,BCHK91], which focusses more specifically on the distributed aspects of systems. At first sight, the concepts of causality and *distribution in space* may appear as dual notions, which should give rise to the same kinds of noninterleaving semantics for distributed systems. In fact this is not the case, essentially because communication may introduce causal dependencies between activities at different locations. In a distributed view, such

*This work has been supported by the ESPRIT/BRA CEDISYS project.

A Theory of Processes with Localities

Une théorie des processus répartis

Gérard Boudol & Ilaria Castellani

INRIA Sophia-Antipolis

Matthew Hennessy

CSAI, University of Sussex

Astrid Kiehn

TUM, Munich

Abstract.

We study a notion of observation for concurrent processes which allows the observer to see the distributed nature of processes, giving explicit names for the location of actions. A general notion of bisimulation related to this observation of distributed systems is introduced. Our main result is that these bisimulation relations, particularized to a process algebra extending CCS, are completely axiomatizable. We discuss in details two instances of location bisimulations, namely the location equivalence and the location preorder.

Résumé.

Nous étudions une notion d'observation des processus concurrents dans laquelle la structure de répartition apparait, sous la forme de noms de sites attribués aux actions. Nous introduisons une notion générale de bisimulation relative a cette observation des systèmes distribués. Notre principal résultat est que l'on peut donner une axiomatisation complète de ces bisimulations, appliquées à une algèbre de processus qui contient CCS. Nous discutons deux cas particuliers de telles bisimulations, que nous appelons équivalence et préordre de répartition.

"cross-causalities" induced by communication are observed as purely temporal dependencies, while in the causal approach they are assimilated to "local causalities", that is causalities induced by sequential composition. Let us see give example, using a CCS notation for processes. Let $\approx_c$ denote a causal equivalence such as the causal bisimulation of [DD89], which turns out to coincide with the weak version of "history preserving bisimulation" [vGG90] and with an instance of "NMS equivalence" [DDNM87]. Now, if $\approx_d$ denotes a distributed bisimulation such as that of [CH89,Cas88,Kie89] we have, for $r = (a.\alpha + b.\beta \mid \bar{\alpha}.b + \bar{\beta}.a)$:

$$\begin{array}{ccccc} r + (a \mid b) & \approx_d & r & \not\approx_d & r + a.b \\ & \not\approx_c & & \approx_c & \end{array}$$

If we add a restriction around r to prevent actions $\alpha, \bar{\alpha}, \beta, \bar{\beta}$ from occurring independently, these absorptions give rise to the following identifications, which are perhaps more suggestive:

$$a \mid b \quad \approx_d \quad (a.\alpha + b.\beta \mid \bar{\alpha}.b + \bar{\beta}.a)\backslash\alpha,\beta \quad \approx_c \quad a.b + b.a$$

Of course in both the causal and the distributed approach $a \mid b$ is distinguished from $a.b + b.a$.

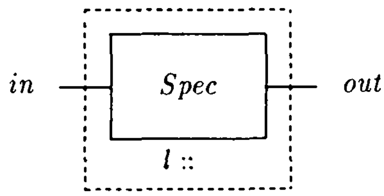
In this paper we develop a semantic theory for CCS which takes the distributed nature of processes into account, along the lines of the above-mentioned works [CH89,Cas88,Kie89]. The first two of these papers do not deal with the restriction operator of CCS, and one may argue that the treatment of restriction in [Kie89] is not completely satisfactory. Here we use a different formalisation, similar to that of [BCHK91]: we shall deal with processes with explicit localities or *locations*, extending CCS with a construct of *location prefixing* $l :: p$, which denotes the process p residing at location l. Intuitively, locations will serve to distinguish different parallel components. Let us illustrate our approach with a more concrete example. We may describe in CCS a simple protocol, transferring data one at a time from one port to another, as follows:

$$\begin{array}{rcl} Sys & \Leftarrow & (Sender \mid Receiver)\backslash\alpha,\beta \\ Sender & \Leftarrow & in.\bar{\alpha}.\beta.Sender \\ Receiver & \Leftarrow & \alpha.out.\bar{\beta}.Receiver \end{array}$$

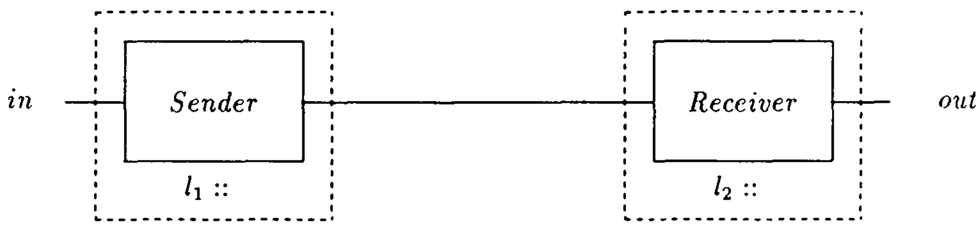
where α represents transmission of a message from the sender to the receiver, and β is an acknowledgement from the receiver to the sender, signalling that the last message has been processed. In the standard theory of *weak bisimulation equivalence*, usually noted $\approx$, one may prove that this system is equivalent to the following specification:

$$Spec \;\Leftarrow\; in.\,out.\,Spec$$

That is to say, $Spec \approx Sys$. The reader familiar with [DD89] should also be readily convinced that $Spec \approx_c Sys$: intuitively, this is because the synchronizations on α, β in Sys create "cross-causalities" between its visible actions *in* and *out*, constraining them to happen alternately in sequence. On the other hand *Spec* will be distinguished from *Sys* in our theory, because *Spec* is completely sequential and thus performs the actions *in* and *out* at the same location l, what can be represented graphically as follows:



while *Sys* is a system distributed among two different localities l_1 and l_2, with the actions *in* and *out* occurring at l_1 and l_2 respectively. Thus *Sys* may be represented as:



Here the unnamed link represents the communication lines α, β, which are private to the system. Although *Spec* and *Sys* will not be equated in our theory, we will be interested in relating them by a weaker relation, a *preorder* that orders processes according to their degree of distribution. A similar "concurrency refinement" preorder, based on causality rather than distribution, was proposed by L. Aceto in [Ace89] for a subset of CCS.

Consider another example, taken from [BCHK91], describing the solution to a simple mutual exclusion problem. In this solution, two processes compete for a device, and a semaphore is used to serialize their accesses to this device:

$$\begin{aligned} Proc &\;\Leftarrow\; \overline{p}.\,enter.\,exit.\,v.\,Proc \\ Sem &\;\Leftarrow\; p.\,\overline{v}.\,Sem \\ Sys &\;\Leftarrow\; (Proc \mid Sem \mid Proc)\backslash\{p, v\} \end{aligned}$$

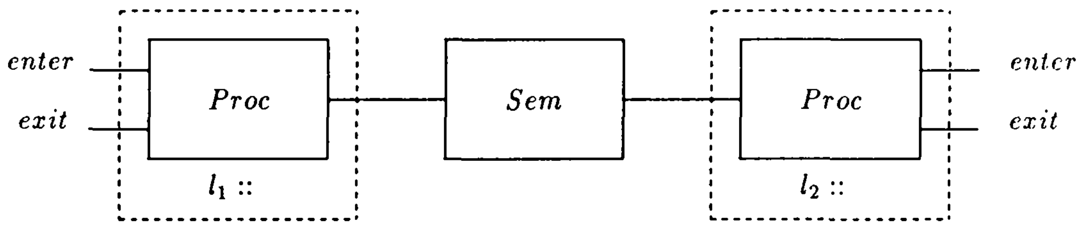
Take now a variant of the system *Sys*, where one of the processes is faulty and may deadlock after exiting the critical region (the deadlocked behaviour is modelled here as *nil*). This system, *FSys*, may be defined by:

$$\begin{aligned} FProc &\Leftarrow \overline{p}.\,enter.\,exit.(v.\,FProc + v.\,nil) \\ FSys &\Leftarrow (Proc \mid Sem \mid FProc)\backslash\{p, v\} \end{aligned}$$

In the standard theory of weak bisimulation the two systems *Sys* and *FSys* are equivalent. In fact they are both equivalent to the sequential specification:

$$Spec \Leftarrow enter.\,exit.\,Spec$$

that is $Sys \approx Spec \approx FSys$. Note that both *Sys* and *FSys* are globally deadlock-free. On the other hand *FSys* has the possibility of entering a *local deadlock* in its faulty component, which has no counterpart in *Sys*. More precisely, consider the following distributed representation of *Sys*:



The faulty system *FSys* has a similar representation, with *FProc* in place of the second occurrence of *Proc*. In this distributed view *Sys* and *FSys* have different behaviours, because *FSys* may reach a state in which no more actions can occur at location l_2, while this is not possible for *Sys*. Note that again the causal approach would make no difference between *Sys* and *FSys*: it may be easily checked that $Sys \approx_c Spec \approx_c FSys$.

In the rest of this introduction, we present our formalisation of distributed systems as CCS processes with explicit *locations*. Let us be more precise about the nature of locations, and the way they are assigned to processes. Since the distributed structure of a system may evolve dynamically (because of the nesting of parallelism and prefixing in CCS terms), the notion of location will have to be structured itself. For example a process of the form $a.\,p$ will be initially considered as having just one location l. Suppose now that $p = q \mid r$, for some nontrivial processes q and r. Then the locations of the subprocesses q and r should be distinguished, to reflect the fact that q and r are independent components; at the same time, they should be both *sublocations* of the location l. In our formalisation we will take locations to be words $u, v, \ldots$ over atomic locations $l, l', \ldots$, and define v to be a sublocation of u whenever u is a prefix of v. Then the subprocesses q and r in the above example will have locations $l\,l_1$, resp. $l\,l_2$, for some atomic locations l_1 and l_2.

As regards the attribution of locations to CCS processes, there are at least two possibilities. The most intuitive approach consists in assigning locations *statically* to the components of a process, using the construct $u :: p$ along the lines suggested by the examples above. Such static assignment has been studied by L. Aceto [Ace91] for the so-called nets of automata, a subset of CCS where parallelism may only appear at the top level. Here we will adopt the different approach of [BCHK91], where locations are *dynamically* generated as the execution – or the observation – of a process goes on. The two approaches are in some sense equivalent (see [Ace91]), though the dynamic view is more convenient for developing technical results, as we shall establish here. In both views, a process with locations is described operationally as performing *location transitions* of the form:

$$p \xrightarrow[u]{a} p'$$

which differ from the standard transitions of CCS in that any (observable) action a has associated with it a particular location u. In the dynamic approach adopted here, these locations are introduced when actions are executed. The essence of our semantics is expressed by the transition rules for action prefixing $a.p$ and location prefixing $u :: p$. The rules for the remaining operators of CCS are formally identical to the standard ones (with $\xrightarrow[u]{a}$ replacing $\xrightarrow{a}$). The rule for action prefixing is:

$$a.p \xrightarrow[l]{a} l :: p \qquad \text{for any atomic location } l$$

This says that the process $a.p$ may be observed to perform an action a at any atomic location l. All subsequent actions of the process will be observed within this location: this is expressed by the fact that the residual of $a.p$ is $l :: p$, the process p residing at location l. The rule for the location prefixing operator $u :: p$ is now:

$$p \xrightarrow[v]{a} p' \quad \Rightarrow \quad u :: p \xrightarrow[uv]{a} u :: p'$$

Thus any action of $u :: p$ is observed at a sublocation of u. Note that the process $u :: p$ retains the location u throughout its execution, in other words location prefixing is a static construct.

We will mainly be interested in the *weak location transition system* associated with the transitions $\xrightarrow[u]{a}$. We assume that unobservable τ–actions have also unobservable locations:

thus τ–transitions will have the usual form $\xrightarrow{\tau}$, and simply pass over existing locations without introducing any new ones.

In a previous paper [BCHK91] we used a similar notion of location transition system to give a non-interleaving semantics for CCS, by extending the notion of bisimulation. The work presented here differs from that of [BCHK91] in two respects. First, the transition rule given here for action prefixing is slightly different, in that it associates with an initial action an atomic location l instead of a general location u. We shall see in section 5 that this difference is significant, and that the semantics proposed here is more in line with our intuition about spatial distribution. Second, we shall be interested now in a more general notion of bisimulation on location transition systems, which we call *parameterized location bisimulation.*

A parameterized location bisimulation *(plb)* is a relation $\mathcal{B}(R)$ on processes with locations, parameterized on a relation R on locations. Roughly speaking, two processes are related by $\mathcal{B}(R)$ if they can perform the same actions at locations u, v related by R. Our main result is a *complete axiomatisation*, over the set of finite CCS processes, of parameterized location bisimulations satisfying some general conditions, which we call *sensible.* This is achieved by introducing an auxiliary prefixing construct $< a \text{ at } ux >.\, p$. Intuitively, the construct $< a \text{ at } ux >.\, p$ prefixes the term t by an "action with locality". Here u represents the access path to the component performing the action a, while x is a location variable that is instantiated to some actual location l when the action is performed. The operational behaviour of such a process is given by the rule:

$$< a \text{ at } ux >.\, p \xrightarrow[ul]{a} p[l/x]$$

This prefixing construct is used to define *normal forms*, which are terms of the form $\sum_{i \in I} < a_i \text{ at } u_i x_i >.\, p_i + \sum_{j \in J} \tau.\, q_j$, and an essential part of our proof system for sensible *plb*'s consists of laws for converting terms into such normal forms. For instance a basic law is the following, which is used to replace ordinary prefixing by the new prefixing construct:

$$a.\, p \;=\; < a \text{ at } x >.\, x :: p$$

We shall study in some detail two instances of sensible parameterized location bisimulation, the *location equivalence* $\approx_\ell$ and the *location preorder* $\sqsubseteq_\ell$. Location equivalence is obtained by taking the relation R on locations to be the identity: then two processes are equivalent when they can perform the same actions at the same locations. The equivalence $\approx_\ell$ formalises the idea that two processes are bisimilar, in the classical sense, and moreover they have the same parallel structure. We will compare the relation $\approx_\ell$ with the earlier version proposed in [BCHK91], and show that $\approx_\ell$ is a stronger notion. We shall also compare $\approx_\ell$ with the distributed bisimulation equivalence $\approx_d$ of [CH89,Cas88,Kie89] and show that the two notions are very close, in that they coincide on finite restriction-free processes.

The other example of sensible *plb* we shall consider, the location preorder $\sqsubseteq_\ell$, relates two processes when they are bisimilar but one is possibly less distributed than the other. The relation $\sqsubseteq_\ell$ is weaker than $\approx_\ell$, in the sense that $\approx_\ell \subset \sqsubseteq_\ell$. For instance, looking back at the protocol example of p.3, we will have the following relations between the specification *Spec* and the protocol *Sys*:

$$Spec \not\approx_\ell Sys \qquad \text{but} \qquad Spec \sqsubseteq_\ell Sys$$

whereas the two systems Sys and $FSys$ in the mutual exclusion example of p.4 are distinguished by both $\approx_\ell$ and $\sqsubseteq_\ell$.

To conclude this introduction, let us say a few words about related work. We already mentioned the work by L. Aceto [Ace91], which provides static characterisations of the relations $\approx_\ell$ and $\sqsubseteq_\ell$ for a general class of CCS processes, the so-called nets of automata. This is interesting not only from an intuitive point of view, but also because it yields an effective version of our theory (the reader could have noticed that the location transition system determined by the simple process $a.\,nil$ is infinitely branching). The notion of explicit locality is used by A. Kiehn in [Kie91] to bring together in the same framework the causal and distributed views of concurrent systems. A similar idea motivates some recent work by U. Montanari and D. Yankelevich [MY91], where the notion of locality is extracted from the proofs of transitions. Their approach provides another effective version of the location equivalence for behaviourally finite processes – but not in general for regular systems, such as the nets of automata.

2 Parameterized Location Bisimulations

In this section we introduce a new kind of transition system, called the *location transition system*, to specify processes whose actions may occur at different *locations*. Let us explain the intuition for the location transition system. The general idea is that processes consist of parallel components which reside at different locations and thus may be observed independently. Then instead of assuming a single global observer for a system we assume a set of observers, one for each parallel component. At each stage of evolution of the system, an observer – or parallel component – has a current *location*, which we represent here as a word u over a set of atomic locations *Loc*. This location may be seen as the "access path" to that component. In this section we are not concerned with the way these access paths are generated; we simply assume that they exist.

Let us now define our transition system, formalising the notion of process with locations. We assume an infinite set of atomic locations *Loc*, ranged over by $k, l, m \ldots$; we then define general *locations*, ranged over by $u, v, w \ldots$, to be sequences of Loc^*. As usual we denote concatenation by uv, and the empty word by ε. The set of non-empty locations is Loc^+. Processes will have transitions $p \xrightarrow[u]{a} p'$, where a is an action and u is the location where it occurs, as well as unobservable τ-transitions; the locations of τ-transitions are themselves considered to be unobservable, so these transitions will have the usual form $p \xrightarrow{\tau} p'$.

Definition 2.1 A *Location Transition System* is of the form

$$(S, A, Loc, \{ \xrightarrow[u]{a} \mid a \in A,\ u \in Loc^*\}, \xrightarrow{\tau})$$

where S is a set of *processes with locations*, A is a set of actions, Loc is the set of atomic locations and each $\xrightarrow[u]{a}$, $\xrightarrow{\tau}$ is a subset of $(S \times S)$, called an action relation. The union of action relations forms the transition relation over S. □

Based on the transitions $p \xrightarrow[u]{a} p'$ and $p \xrightarrow{\tau} p'$, we define the weak transitions $p \overset{\varepsilon}{\Rightarrow} p'$ and $p \overset{a}{\underset{u}{\Longrightarrow}} p'$ in the standard way: we let $\overset{\varepsilon}{\Rightarrow} = (\xrightarrow{\tau})^n,\ n \geq 0$. We will also use $\overset{\tau}{\Rightarrow}$ to denote $(\xrightarrow{\tau})^n,\ n \geq 1$. Then the $\overset{a}{\underset{u}{\Longrightarrow}}$ are given by:

$$p \overset{a}{\underset{u}{\Longrightarrow}} p' \ \Leftrightarrow_{\text{def}}\ \exists q, q'.\ p \overset{\varepsilon}{\Rightarrow} q \ \xrightarrow[u]{a}\ q' \overset{\varepsilon}{\Rightarrow} p'$$

On the resulting (weak) location transition system we define now the notion of *parameterized location bisimulation (plb)*. A *plb* is a relation on processes with locations, parameterized on a relation R on locations. Informally, two processes are related if they can perform the same actions, at locations u, v related by R. Intuitively, R is a requirement on the way corresponding transitions should be reached in related processes.

Definition 2.2 Let $R \subseteq (Loc^* \times Loc^*)$ be a relation on locations. A relation $G \subseteq (S \times S)$ is a *parameterized location bisimulation (plb)* parameterized on R, or R-location bisimulation, iff $G \subseteq C_R(G)$, where $(p, q) \in C_R(G)$ iff

(i)	$p \overset{\varepsilon}{\Rightarrow} p'$	implies	$q \overset{\varepsilon}{\Rightarrow} q'$	for some $q' \in S$ such that $(p', q') \in G$
(ii)	$q \overset{\varepsilon}{\Rightarrow} q'$	implies	$p \overset{\varepsilon}{\Rightarrow} p'$	for some $p' \in S$ such that $(p', q') \in G$
(iii)	$p \overset{a}{\underset{u}{\Longrightarrow}} p'$	implies	$q \overset{a}{\underset{v}{\Longrightarrow}} q'$	for some $q' \in S$ and $v \in Loc^*$ such that $(u, v) \in R$ and $(p', q') \in G$
(iv)	$q \overset{a}{\underset{v}{\Longrightarrow}} q'$	implies	$p \overset{a}{\underset{u}{\Longrightarrow}} p'$	for some $p' \in S$ and $u \in Loc^*$ such that $(u, v) \in R$ and $(p', q') \in G$.

□

The function C_R is monotonic and therefore, from standard principles, it has a maximal fixpoint which we denote by $\mathcal{B}(R)$. As usual

$$\mathcal{B}(R) = \bigcup\{ G \mid G \subseteq C_R(G) \}$$

Other properties of $\mathcal{B}(R)$ depend on corresponding properties of the underlying relation R. For instance we have:

Property 2.3 *If R is reflexive (resp. symmetric, transitive) then so is $\mathcal{B}(R)$.*

Proof. Straightforward. □

It should be clear that if $R \subseteq R'$ then any R-location bisimulation is also an R'-location bisimulation, therefore:

Property 2.4 $\quad R \subseteq R' \;\Rightarrow\; \mathcal{B}(R) \subseteq \mathcal{B}(R')$

If for instance we take R to be the universal relation $U = Loc^* \times Loc^*$, we obtain an equivalence relation, $\mathcal{B}(U)$, which is the largest parameterized location bisimulation. Intuitively, letting $R = U$ amounts to ignore the information on locations. In the next section we will see that indeed for the location transition system associated with the language CCS the relation $\mathcal{B}(U)$ coincides with the standard *weak bisimulation equivalence* of [Mil89].

Another important instance of parameterized location bisimulation is $\mathcal{B}(Id)$, where Id is the identity relation on locations. Again this is an equivalence relation, which we shall call *location equivalence* and denote by $\approx_\ell$. This equivalence, which equates processes with the same degree of distribution, will be studied in detail in sections 4 and 5. We shall see that in some sense location equivalence is the strongest "reasonable" parameterized location bisimulation. In section 6 we will discuss another example of parameterized location bisimulation, the *location preorder* $\sqsubseteq\!\!\!\!\raisebox{-1ex}{$\approx$}_\ell$, a preorder formalising the idea that a process is less distributed than another.

3 Language and Operational Semantics

We propose now a location transition system semantics for an extension of Milner's language CCS, and discuss the resulting parameterized location bisimulations. We should point out that the semantics presented here is very similar, but not identical, to the one given in [BCHK91]. The reasons for introducing a new, more discriminating semantics are both technical and intuitive; they will be explained in the next sections.

The language we consider is essentially CCS, with some additional constructs to deal with locations. As usual we assume a set of actions of the form $Act = \Lambda \cup \bar{\Lambda}$, where Λ is a set of names ranged over by $\alpha, \beta, \ldots$, $\bar{\Lambda}$ the corresponding set of co-names $\{\bar{\alpha} \mid \alpha \in \Lambda\}$, and $\bar{\ }$ is a bijection such that $\bar{\bar{\alpha}} = \alpha$ for all $\alpha \in \Lambda$. The symbol τ, not belonging to Act, denotes the invisible action. We use $a, b, c, \ldots$ to range over Act and $\mu, \nu, \ldots$ to range over $Act_\tau = Act \cup \{\tau\}$. We will use $p, q, \ldots$ to denote terms of our language. The set of process variables, ranged over by $P, Q \ldots$, is denoted $PVar$. The operators we consider are all those of CCS, namely nil, action prefixing $\mu. p$, nondeterministic choice $+$, parallel composition $|$, relabelling $[f]$, restriction $\backslash\alpha$ and recursion $rec\, P.\ p$. In addition we shall use the construct of *location prefixing* $u :: p$ (already introduced in [BCHK91]) to represent an agent p residing at the location u. We recall from the previous section that locations $u, v, w \ldots$ are words of Loc^*.

Moreover we shall assume, for axiomatization purposes, an infinite set of *location variables* $LVar$, ranged over by $x, y \ldots$, and introduce a new form of prefixing, $< a \text{ at } \sigma x >. p$, where σ is a location word possibly containing variables, that is $\sigma \in (Loc \cup LVar)^*$. Intuitively, the construct $< a \text{ at } \sigma x >. p$ prefixes a term by an "action with locality". The meaning of this operator will be specified more precisely when we give the formal semantics of our language. Because of location variables, we will need a more general location construct of the form $\sigma :: p$, where $\sigma \in (Loc \cup LVar)^*$; thus $u :: p$ will be a particular case of $\sigma :: p$. To sum up, our language $\mathbb{L}$ is given by:

$$\begin{array}{rcl} p & ::= & nil \mid \mu.p \mid p+p \mid p \,|\, p \mid p[f] \mid p\backslash\alpha \\ & & \mid P \mid rec\, P.\ p \\ & & \mid \sigma :: p \mid < a \text{ at } \sigma x >. p \end{array}$$

Here $rec\, P.\ p$ is a binding operator for process variables, which leads to the usual definition of free and bound occurrences of variables and of substitution $[p/P]$ of terms for process variables. Similarly, $< a \text{ at } \sigma x >. p$ is a *binding operator* for location variables, which binds all free occurrences of the variable x in p. However, x may still occur free in σ. Once more, this leads to standard definitions of free and bound occurrences of location variables and of substitution $[u/x]$ of locations for location variables. We will use the notation $p[\rho]$ to denote an instantiation of both process and location variables in p. Similarly, we shall use $\sigma[\rho]$ to represent an instantiation of an "open" location word σ. Thus we have for example: $(< a \text{ at } \sigma x >. p)[\rho] = < a \text{ at } \sigma[\rho] y >. (p[y/x])[\rho]$ where y is a fresh variable. In general we will be only interested in *closed terms*, where all occurrences of both kinds of variables are bound. We take $\mathbb{P}$ to denote the set of such closed terms, also called *processes* in the following. We shall still use $p, q \ldots$ to range over $\mathbb{P}$, specifying whether we deal with closed or open terms when this is not clear from the context. Note that if $< a \text{ at } \sigma x >. p$ is a process then σ must be a word over location names only, i.e. $\sigma \in Loc^*$. For any process p, we shall denote by $loc(p)$ the set of location names $l \in Loc$ occurring in p. The set of *finite* processes, that is those not involving the recursion construct, will be denoted $\mathbb{P}_f$.

We define now the location transition system for $\mathbb{P}$, specifying its operational semantics. The transition rules are given in Figure 1. As we said in the previous section, the idea is that actions are observed at particular locations. Initially, some locations may be present in processes because of the location construct $u :: p$. Pure CCS processes contain no locations: one may regard them as having all components at the empty location ε. Subsequently, when an action is performed by a component at some location u, an atomic location l is created, which is appended to u to form the new location ul. The word u may then be understood as the *access path* to the component performing the action. For the prefixing operator $a.p$ of CCS the "access path" is empty, and we have the following transition rule:

$$a.p \xrightarrow[l]{a} l :: p \quad \text{for any atomic location } l \in Loc$$

Here the action a may be observed at an arbitrary location $l \in Loc$. The only difference with the semantics given in [BCHK91] is that here the location where the action occurs is atomic, i.e. it is a letter l of Loc instead of a word u of Loc^*.

For the new prefixing construct $< a \text{ at } ux >. p$ the access path is given by u, while x is a variable which is replaced by an arbitrary location l when a is executed. Thus the rule for this operator is:

$$< a \text{ at } ux >. p \xrightarrow[ul]{a} p[l/x] \quad \text{for any atomic location } l \in Loc$$

Note that for $u = \varepsilon$ and $p = x :: q$ the process $< a \text{ at } ux >. p$ has the same behaviour as $a.q$:

$$< a \text{ at } x >. x :: q \xrightarrow[l]{a} l :: q \quad \text{for any atomic location } l \in Loc$$

The remaining rules of Figure 1 are modelled on the standard ones for CCS. They are exactly the same as those in [BCHK91]. For example $p + q$ can perform any of the moves of either p or q while $u :: p$ has all the moves of p with locations prefixed by u.

For each $a \in Act$ let $\xrightarrow[u]{a} \subseteq (\mathbb{P} \times \mathbb{P})$ be the least binary relation satisfying the following axioms and rules.

(LT1)	$a.p \xrightarrow[l]{a} l :: p$	$l \in Loc$	
(LT2)	$< a \text{ at } ux >.p \xrightarrow[ul]{a} p[l/x]$	$l \in Loc$	
(LT3)	$p \xrightarrow[u]{a} p'$	implies	$v :: p \xrightarrow[vu]{a} v :: p'$
(LT4)	$p \xrightarrow[u]{a} p'$	implies	$p + q \xrightarrow[u]{a} p'$ $q + p \xrightarrow[u]{a} p'$
(LT5)	$p \xrightarrow[u]{a} p'$	implies	$p \mid q \xrightarrow[u]{a} p' \mid q$ $q \mid p \xrightarrow[u]{a} q \mid p'$
(LT6)	$p \xrightarrow[u]{a} p'$	implies	$p[f] \xrightarrow[u]{f(a)} p'[f]$
(LT7)	$p \xrightarrow[u]{a} p'$	implies	$p\backslash\alpha \xrightarrow[u]{a} p'\backslash\alpha,\ a \notin \{\alpha, \bar{\alpha}\}$
(LT8)	$p[rec\,P.\ p/P] \xrightarrow[u]{a} p'$	implies	$rec\,P.\ p \xrightarrow[u]{a} p'$

Figure 1: Location transitions for $\mathbb{P}$

For each $\mu \in Act_\tau$ let $\xrightarrow{\mu} \subseteq (\mathbb{P} \times \mathbb{P})$ be the least binary relation satisfying the following axiom and rules.

(ST1)	$\mu.p \xrightarrow{\mu} p$		
(ST2)	$< a \text{ at } ux >.p \xrightarrow{a} p[x\surd]$		
(ST3)	$p \xrightarrow{\mu} p'$	implies	$u :: p \xrightarrow{\mu} u :: p'$
(ST4)	$p \xrightarrow{\mu} p'$	implies	$p + q \xrightarrow{\mu} p'$ $q + p \xrightarrow{\mu} p'$
(ST5)	$p \xrightarrow{\mu} p'$	implies	$p \mid q \xrightarrow{\mu} p' \mid q$ $q \mid p \xrightarrow{\mu} q \mid p'$
(ST6)	$p \xrightarrow{\mu} p'$	implies	$p[f] \xrightarrow{f(\mu)} p'[f]$
(ST7)	$p \xrightarrow{\mu} p'$	implies	$p\backslash\alpha \xrightarrow{\mu} p'\backslash\alpha,\ \mu \notin \{\alpha, \bar{\alpha}\}$
(ST8)	$p[rec\, P.\ p/P] \xrightarrow{\mu} p'$	implies	$rec\, P.\ p \xrightarrow{\mu} p'$
(ST9)	$p \xrightarrow{a} p', \quad q \xrightarrow{\bar{a}} q'$	implies	$p \mid q \xrightarrow{\tau} p' \mid q'$

Figure 2: Standard Transitions for $\mathbb{P}$

By inspecting the rules one can easily check the following property:

$$p \xrightarrow[v]{a} p' \Rightarrow \exists u \in loc(p)^* \ \exists l \in Loc. \quad v = ul$$

Thus in what follows we will often write transitions explicitly in the form $p \xrightarrow[ul]{a} p'$, and refer to u as the "access path", and to l as the "actual location" of the action a. One may show the following property, stating that the actual location l can be chosen arbitrarily at each step:

Property 3.1 *For any term p and L such that $loc(p) \subseteq L \subset Loc$, if $p \overset{a}{\underset{ul}{\Longrightarrow}} p'$ then $\forall k \notin L\ \exists p''.\ p \overset{a}{\underset{uk}{\Longrightarrow}} p''$ and $p''[k \to l] = p'$, and $p \overset{a}{\underset{uh}{\Longrightarrow}} p''[k \to h]$ for any $h \in Loc$.*

The transitions $p \xrightarrow{\tau} p'$, whose location is not observable, are defined through a simple adaptation of the standard transition system for CCS to our extended language, which is described in Figure 2. The only new rules are the ones for the constructs $u :: p$ and $< a \text{ at } ux >.p$; in these rules the locations are in fact ignored. In particular, for the second construct we use the notation $p[x\surd]$ to represent the term p where all free occurrences of x have been erased, that is $p[x\surd] = p[\varepsilon/x]$; for instance $(x :: p)[x\surd] = \varepsilon :: (p[x\surd])$, and $(< a \text{ at } xx >.p)[x\surd] = < a \text{ at } x >.p$ because the first occurrence of x is free and the second occurrence of x binds this variable in p. The weak transitions $p \overset{a}{\underset{u}{\Longrightarrow}} p'$ are then derived as explained in the previous section.

In the last two sections of the paper, we shall consider specific subsets of $\mathbb{P}$ that consist of terms representing *nets of agents*. These terms are built on top of given agents using the static constructs of the language. More precisely, given a subset Ag of $\mathbb{P}$, we denote by $\mathbb{N}(Ag)$ the set of terms given by the following grammar:

$$r ::= p \quad | \quad u :: r \quad | \quad (r \mid r) \quad | \quad r[f] \quad | \quad r \backslash a$$

where p is any process of Ag. This syntax extends Milner's one for flowgraphs (see [Mil79]). The same syntax is used by Aceto [Ace91] to define what he calls "states", which include the nets of automata. Obviously this $\mathbb{N}(Ag)$ is only interesting for a set Ag of agents which is not closed for the static constructs. For instance if we take the CCS processes as agents, then the terms $a.(u :: p)$ and $p + u :: q$ are not in $\mathbb{N}(\text{CCS})$. We shall also use the notation $\mathbb{N}_r(Ag)$ for the set of terms built on top of agents of Ag using the static constructs except restriction. It is easy to see that the static structure of the nets is preserved by transitions, that is:

Lemma 3.2 *Let Ag be a subset of $\mathbb{P}$ closed w.r.t. transitions, that is satisfying*
(i) *if $p \in Ag$ and $p \xrightarrow{\mu} p'$ then $p' \in Ag$*
(ii) *if $p \in Ag$ and $p \xrightarrow[u]{a} p'$ then $p' \in Ag$*
Then $\mathbb{N}(Ag)$ is closed w.r.t. transitions.

$$
\begin{array}{lrcl}
\text{(SL1)} & r \mid s & = & s \mid r \\
\text{(SL2)} & r \mid (s \mid q) & = & (r \mid s) \mid q \\
\text{(SL3)} & p & = & \varepsilon :: p \\
\text{(SL4)} & u :: (r \mid s) & = & u :: r \mid u :: s \\
\text{(SL5)} & u :: (v :: r) & = & uv :: r \\
\text{(SL6)} & (r \mid s)[f] & = & r[f] \mid s[f] \\
\text{(SL7)} & (u :: r)[f] & = & u :: (r[f]) \\
\text{(SL8)} & (u :: r)\backslash b & = & u :: (r\backslash b)
\end{array}
$$

Figure 3: Some Static Laws

Obviously the same result holds for $\mathbb{N}_r(Ag)$. Sometimes it will be useful to abstract to some extent from the static structure of a net, by considering it up to some reorganization that preserves transitions. More precisely let $\equiv$ be the congruence over $\mathbb{N}(Ag)$ (regarded as the algebra of terms generated by Ag using the static constructs) induced by the equations SL1-SL8 given in Figure 3. Then it is easy to show that $\equiv$ is a "strong (Id-location) bisimulation" that is:

Lemma 3.3 *Let Ag be a subset of $\mathbb{P}$ closed w.r.t. transitions. Then for any $p, q \in \mathbb{N}(Ag)$*
(i) *if $p \equiv q$ and $p \xrightarrow{\mu} p'$ then there exists q' such that $q \xrightarrow{\mu} q'$ and $p' \equiv q'$*
(ii) *if $p \equiv q$ and $p \xrightarrow[u]{a} p'$ then there exists q' such that $q \xrightarrow[u]{a} q'$ and $p' \equiv q'$.*

The proof is left as an exercise. □

We establish now a result that will be used in the next section, which relates the location transitions and the operation of substitution of locations for location variables, denoted $p[\rho]$.

Lemma 3.4 *For any term p:*

1) $p\,[\rho] \overset{\varepsilon}{\Rightarrow} p'$ *implies* $\exists\, p''$ *such that* $p' = p''\,[\rho]$ *and* $\forall \rho'.\ p[\rho'] \overset{\varepsilon}{\Rightarrow} p''\,[\rho']$

2) $p\,[\rho] \underset{ul}{\overset{a}{\Rightarrow}} p'$ *implies* $\exists\, \sigma, p''.\ u = \sigma\,[\rho],\ p' = p''\,[\rho]$ *and* $\forall \rho'.\ p[\rho'] \xRightarrow[\sigma\,[\rho']l]{a} p''\,[\rho']$

Proof. We prove the first point by induction on the definition of $p\,[\rho] \overset{\varepsilon}{\Rightarrow} p'$. Clearly it is enough to prove this statement for "strong" arrows. More precisely, we show

$$p\,[\rho] \xrightarrow{\mu} p' \;\Rightarrow\; \exists\, p''.\, p' = p''\,[\rho]\ \&\ \forall \rho'.\ p[\rho'] \xrightarrow{\mu} p''\,[\rho']$$

by induction on the definition of the transition. The case of ST2 is the only one deserving some consideration. If $p[\rho] = \,< a \text{ at } ux >.q$ then there exist σ and r such that $p = \,< a \text{ at } \sigma z >.r$

with $u = \sigma[\rho]$ and $q = r[x/z][\rho]$ where x does not occur free in r, and is not affected by ρ. Then $\mu = a$ and $p' = q[x\surd] = (r[z\surd])[\rho]$. For any ρ' we have $p[\rho'] = < a \text{ at } \sigma[\rho']y >. r[y/z][\rho']$ for some fresh variable y, and $p[\rho'] \xrightarrow{a} (r[y/z][\rho'])[y\surd] = (r[z\surd])[\rho']$, therefore we may let $p'' = r[z\surd]$.

Regarding the second point, we have by definition $p[\rho] \overset{a}{\underset{ul}{\Longrightarrow}} p'$ if and only if there exist p_0 and p_1 such that $p[\rho] \overset{\varepsilon}{\Rightarrow} p_0 \xrightarrow[ul]{a} p_1 \overset{\varepsilon}{\Rightarrow} p'$. Then, using the previous point, we only have to prove the statement for "strong" transitions $p[\rho] \xrightarrow[ul]{a} p'$. One proceeds by induction on the inference of this transition. We omit the proofs. Just note that the case of LT2 is very similar to the case of ST2 in point 1), and that in the case of LT3 we have $p[\rho] = v :: q$, therefore $p = \sigma :: r$ with $v = \sigma[\rho]$ and $q = r[\rho]$. □

We may now instantiate definition of parameterized location bisimulation to obtain a family of relations $\mathcal{B}(R)$ over $\mathbb{P}$. These relations are extended to open terms in the standard way: for terms p, q involving process and location variables we set $p\,\mathcal{B}(R)\,q$ if $p[\rho]\,\mathcal{B}(R)\,q[\rho]$ for every closed instantiation ρ of both process and location variables.

We already mentioned in the previous section the case where R is U, the universal relation on locations. In $\mathcal{B}(U)$ the locations are completely ignored and therefore one expects it to coincide with the usual (*weak*) *bisimulation equivalence* $\approx$. The bisimulation equivalence $\approx$ is defined on our extended language in the standard way, using the weak transitions $\overset{\mu}{\Rightarrow}$ associated with the transitions $\xrightarrow{\mu}$ of Figure 2. We may then show the following:

Proposition 3.5 *For all processes p, q:* $(p,q) \in \mathcal{B}(U)$ *if and only if* $p \approx q$.

Proof. For any term r let $pure(r)$ be the CCS term obtained by removing all locations from r, for instance $pure(< a \text{ at } \sigma x >. s) = a.pure(s)$ and $pure(\sigma :: s) = pure(s)$. Let now $p, q \in \mathbb{P}$. Obviously $p \overset{\mu}{\Rightarrow} p'$ implies $pure(p) \overset{\mu}{\Rightarrow} pure(p')$, and conversely if $pure(p) \overset{\mu}{\Rightarrow} p'$, then there exists p'' such that $p \overset{\mu}{\Rightarrow} p''$ and $p' = pure(p'')$. Therefore $p \approx pure(p)$ and thus it is sufficient to establish

$$pure(p) \approx pure(q) \text{ if and only if } (p,q) \in \mathcal{B}(U).$$

The proof of this fact depends on relating the two different types of transitions $\overset{a}{\Rightarrow}$ and $\overset{a}{\underset{u}{\Longrightarrow}}$. One can show that

1. if $p \overset{a}{\underset{u}{\Longrightarrow}} p'$ then $pure(p) \overset{a}{\Rightarrow} pure(p')$
2. if $pure(p) \overset{a}{\Rightarrow} r$ then there exist $u \in Loc^*$ and $p' \in \mathbb{P}$ such that $p \overset{a}{\underset{u}{\Longrightarrow}} p'$ and $r = pure(p')$

3. $p \overset{s}{\Rightarrow} p'$ if and only if $pure(p) \overset{s}{\Rightarrow} pure(p')$.

Now let

$$G = \{ (p,q) \mid pure(p) \approx pure(q) \}.$$

Using the previous facts one shows that $G \subseteq C_U(G)$ and therefore $pure(p) \approx pure(q)$ implies $(p,q) \in \mathcal{B}(U)$. Conversely let

$$B = \{ (pure(p), pure(q)) \mid (p,q) \in \mathcal{B}(U) \}.$$

Once more facts 1,2,3 can be used to show that B is a standard bisimulation and therefore $(p,q) \in \mathcal{B}(U)$ implies $p \approx q$. □

We have seen in the previous section that for any R, the relation $\mathcal{B}(R)$ is included into $\mathcal{B}(U)$. Therefore:

Corollary 3.6 *For any relation R and processes p,q:* $(p,q) \in \mathcal{B}(R)$ *implies* $p \approx q$.

We also mentioned the *plb* obtained by taking $R = Id$, the identity relation on locations. This relation, the *location equivalence* $\approx_\ell$, will be studied in detail in Section 5. By Corollary 3.6 we know that this equivalence is at least as discriminating as bisimulation equivalence $\approx$. We give now an example showing that $\approx_\ell$ is strictly finer that $\approx$. Let p and q denote respectively the CCS processes $(a.\alpha.c \mid b.\bar{\alpha}.d)\backslash\alpha$ and $(a.\alpha.d \mid b.\bar{\alpha}.c)\backslash\alpha$. Since in p the actions a and c are in the same parallel component we have $p \overset{a}{\underset{l}{\Rightarrow}} \overset{b}{\underset{k}{\Rightarrow}} \overset{c}{\underset{u}{\Rightarrow}} p' \;\Rightarrow\; u = ll'$ for some l', whereas this is not the case for q. Therefore $p \not\approx_\ell q$, while it is easy to check that $p \approx q$.

Let us consider another example of *plb*, which is a preorder but not an equivalence. Let $\lesssim_{\mathrm{suf}}$ be the *plb* induced by (the inverse of) the suffix relation on words, defined by: $u \, R \, v \Leftrightarrow \exists w$ s.t. $u = wv$, that is v is a *suffix* of u. The relation R is obviously a preorder, and thus the corresponding *plb* $\mathcal{B}(R) = \lesssim_{\mathrm{suf}}$ is also a preorder. This preorder might be used to express the intuition that one process is more sequential or less distributed than another: if p is more sequential than q we expect every move of p to be matched by a move of q from a location which is less nested in the sequential structure of the process and thus somehow "contained" in the location of p. We have for example:

$$a.b.nil + b.a.nil \lesssim_{\mathrm{suf}} a.nil \mid b.nil$$

and more generally

$$a.(p \mid b.q) + b.(a.p \mid q) \lesssim_{\mathrm{suf}} a.p \mid b.q$$

for any processes p, q. However $\lesssim_{\text{suf}}$ is not preserved by location prefixing, and therefore neither by action prefixing. For example if r, s are the processes in one of the above examples then $c.r \not\lesssim_{\text{suf}} c.s$ because this would require, after one execution step, that $l :: r \lesssim_{\text{suf}} l :: s$, which is not true. Essentially $l :: r \not\lesssim_{\text{suf}} l :: s$ because the underlying relation on locations is not preserved by concatenation on the left.

The last examples show that if we want $\mathcal{B}(R)$ to have a reasonable algebraic theory then R must enjoy certain properties. For instance we have the following:

Proposition 3.7 *If R is reflexive and compatible with concatenation on the left, that is $u\,R\,v \;\Rightarrow\; wu\,R\,wv$, then $\mathcal{B}(R)$ is preserved by all the operators in the language except $+$.*

Proof. The reflexivity of R is used in showing that $\mathcal{B}(R)$ is preserved by the prefixing constructs $a.p$ and $< a \text{ at } \sigma x >.p$. The compatibility of R with concatenation plays a role in proving that $\mathcal{B}(R)$ is preserved by location prefixing $u :: p$ and by the prefixing construct $a.p$. We examine these two cases, leaving the others, which follow the standard pattern, to the reader.

Let G be the set $\{(w :: r, w :: s), (a.r, a.s) \mid (r, s) \in \mathcal{B}(R)\}$. Then one can check that $G \subseteq C_R(G)$. For example any possible external move from $w :: r$ must be of the form $w :: r \overset{a}{\underset{wu}{\Longrightarrow}} w :: r'$ where $r \overset{a}{\underset{u}{\Longrightarrow}} r'$. Since $(r, s) \in \mathcal{B}(R)$ there must be a move from s of the form $s \overset{a}{\underset{v}{\Longrightarrow}} s'$ where $u\,R\,v$ and $(r', s') \in \mathcal{B}(R)$. But then the move $w :: s \overset{a}{\underset{wv}{\Longrightarrow}} w :: s'$ matches the original move from $w :: r$ since also $wu\,R\,wv$. This shows that G is an R-location bisimulation, therefore $(p, q) \in \mathcal{B}(R) \;\Rightarrow\; (w :: p, w :: q) \in \mathcal{B}(R)$. □

It should be clear that if R is reflexive, then

$$p \equiv q \;\Rightarrow\; p\,\mathcal{B}(R)\,q$$

Consider now the preorder $\lesssim_{\text{pref}}$ induced by (the inverse of) the *prefix* relation on words, defined by: $u\,R\,v \Leftrightarrow \exists\, w$ s.t. $u = vw$. This relation R is preserved by concatenation on the left and thus $\lesssim_{\text{pref}}$ could be a suitable candidate for our theory. Note that, like the relation $\lesssim_{\text{suf}}$ discussed above, $\lesssim_{\text{pref}}$ could be viewed as a preorder ordering processes according to their "degree of distribution"; however we will see in Section 6 that $\lesssim_{\text{pref}}$ is not completely appropriate to formalise this intuition. Moreover the relation $\lesssim_{\text{pref}}$ lacks another algebraic property that we would like to ensure, namely:

$$uRv \quad\Rightarrow\quad u :: p \;\; \mathcal{B}(R) \;\; v :: p$$

We have for instance $l :: a.\,nil \not\lesssim_{\text{pref}} a.\,nil$. A sufficient condition for this property to hold is the following:

Proposition 3.8 *If R is compatible with concatenation on the right, that is $u\,R\,v \;\Rightarrow\; uw\,R\,vw$, then $\mathcal{B}(R)$ satisfies the property: $uRv \;\Rightarrow\; u::p\;\mathcal{B}(R)\;v::p$*

In what follows we shall also make use of properties of *plb*'s with respect to the operation of *location renaming*. A location renaming is determined by a mapping π from *Loc* to *Loc**, which is extended to words in the obvious way: $\pi(\varepsilon) = \varepsilon$ and $\pi(lu) = \pi(l)\pi(u)$. Further, π is transferred homomorphically to a mapping between processes: for example we have $\pi(u :: p) = \pi(u) :: \pi(p)$ and $\pi(<a \text{ at } ux>.p) = <a \text{ at } \pi(u)x>.\,\pi(p)$. For a renaming affecting only one location we will use the notation $p[l \to u]$, meaning the result of applying to p the renaming π defined by

$$\pi(k) \;=\; \begin{cases} u & \text{if } k = l \\ k & \text{otherwise} \end{cases}$$

Then we have:

Lemma 3.9 *Let $R \subseteq Loc^* \times Loc^*$ be a relation on locations compatible with location renaming, that is $u\,R\,v \;\Rightarrow\; \pi(u)\,R\,\pi(v)$. Then $\mathcal{B}(R)$ is compatible with location renaming on processes, that is $p\,\mathcal{B}(R)\,q \;\Rightarrow\; \pi(p)\,\mathcal{B}(R)\,\pi(q)$.*

Proof. One shows that the relation $\{\; (\pi(p), \pi(q)) \mid p\,\mathcal{B}(R)\,q \;\}$ is an R-location bisimulation. To this end one proves the following properties of transitions w.r.t. location renaming:

1. $\pi(p) \stackrel{\varepsilon}{\Rightarrow} p' \;\Rightarrow\; \exists p''.\; p \stackrel{\varepsilon}{\Rightarrow} p'' \;\&\; p' = \pi(p'')$
2. $p \stackrel{\varepsilon}{\Rightarrow} p' \;\Rightarrow\; \pi(p) \stackrel{\varepsilon}{\Rightarrow} \pi(p')$
3. $\pi(p) \overset{a}{\underset{ul}{\Longrightarrow}} p' \;\Rightarrow\; \exists v\, \forall k \notin loc(p)\, \exists p''.\; p \overset{a}{\underset{vk}{\Longrightarrow}} p'',\; u = \pi(v) \;\&\; p' = \pi'(p'')$ where

$$\pi'(n) \;=\; \begin{cases} l & \text{if } n = k \\ \pi(n) & \text{otherwise} \end{cases}$$

4. $q \overset{a}{\underset{vk}{\Longrightarrow}} q' \;\&\; k \notin loc(q) \;\Rightarrow\; \forall \pi\, \forall l.\; \pi(q) \overset{a}{\underset{\pi(v)l}{\Longrightarrow}} \pi'(q')$ where π' is defined as in the previous point.

The details are left to the reader (see [BCHK91]). □

Since *Id* is obviously compatible with location renaming, we have in particular:

Corollary 3.10 $p \approx_\ell q \;\Rightarrow\; \pi(p) \approx_\ell \pi(q)$

In order to develop an equational theory for parameterized location bisimulations, we need to turn them into substitutive relations, that is relations which are preserved by all the operators of the language. This is done in the standard way. For any *plb* $\mathcal{B}(R)$ we define $\mathcal{B}^c(R)$ to be the closure of $\mathcal{B}(R)$ w.r.t. all contexts:

Definition 3.11 $p\,\mathcal{B}^c(R)\,q$ if and only if for every term context $C[\,]:\ C[p]\,\mathcal{B}(R)\,C[q]$. □

It is a standard result that the relation $\mathcal{B}^c(R)$ so defined is the largest relation compatible with the constructs of the language which is included in $\mathcal{B}(R)$. This relation is a precongruence if R is a preorder, and a congruence if R is an equivalence relation. As usual for weak bisimulations, when the relation R satisfies the hypotheses of the Proposition 3.7, that is R is reflexive and compatible with concatenation on the left, sum-contexts are the only relevant ones in the definition of $\mathcal{B}^c(R)$, and we have the following characterisation:

Property 3.12 *If R is reflexive and compatible with concatenation on the left, then $p\,\mathcal{B}^c(R)\,q$ iff for any action a not occurring in p,q, $p+a\ \mathcal{B}(R)\ q+a$.*

On processes of $\mathbb{P}$, we also have the standard behavioural characterisation for $\mathcal{B}^c(R)$:

Property 3.13 *Let $p,q \in \mathbb{P}$. If R is reflexive and compatible with concatenation on the left, then $p\,\mathcal{B}^c(R)\,q$ if and only if*

1. $p \xrightarrow{\tau} p'$ *implies* $q \overset{\tau}{\Rightarrow} q'$ *for some q' such that* $(p',q') \in \mathcal{B}(R)$
2. $q \xrightarrow{\tau} q'$ *implies* $p \overset{\tau}{\Rightarrow} p'$ *for some p' such that* $(p',q') \in \mathcal{B}(R)$
3. $p \xrightarrow[u]{a} p'$ *implies* $q \underset{v}{\overset{a}{\Longrightarrow}} q'$ *for some q' and v such that* $(p',q') \in \mathcal{B}(R)$ *and* $u\,R\,v$
4. $q \xrightarrow[v]{a} q'$ *implies* $p \underset{u}{\overset{a}{\Longrightarrow}} p'$ *for some p' and u such that* $(p',q') \in \mathcal{B}(R)$ *and* $u\,R\,v$.

We end this section by showing that $\mathcal{B}^c(R)$ is well-behaved w.r.t. the recursion operator.

Proposition 3.14 *If R is a preorder (i.e. reflexive and transitive) and $(s,t) \in \mathcal{B}^c(R)$ then $(rec\,P.\ s, rec\,P.\ t) \in \mathcal{B}^c(R)$.*

Proof. The proof follows the lines of that of Propositions 7.8 and 4.12 of [Mil89] and therefore we only give the outline here. Suppose for convenience that s,t contain no free location variables and no free process variable other than P. Let

$$G = \{\,(r[rec\,P.\ s/P], r[rec\,P.\ t/P]) \mid r \text{ contains at most } P \text{ free }\}.$$

Then one can prove by structural induction on r, as in Proposition 4.12, of [Mil89] that for any $(p,q) \in G$

1. $p \xrightarrow{\tau} p'$ implies $q \stackrel{\tau}{\Rightarrow} q'$ for some q' and p'' such that $p' \, G \, p'' \, \mathcal{B}(R) \, q'$

2. $p \xrightarrow[u]{a} p'$ implies $q \overset{a}{\underset{v}{\Longrightarrow}} q'$ for some q', v and p'' such that $p' \, G \, p'' \, \mathcal{B}(R) \, q'$ and $u \, R \, v$

3. similarly with p and q interchanged.

Since R is transitive this is sufficient to establish that $G \subseteq \mathcal{B}(R)$ and therefore that if $(p,q) \in G$ then $(p,q) \in \mathcal{B}(R)$. By virtue of the characterisation of $\mathcal{B}(R)$ given above, we have now $(p,q) \in \mathcal{B}^c(R)$. If we choose r to be simply P, we have then $(rec\, P.\ s, rec\, P.\ t) \in \mathcal{B}(R)$. □

Another property we expect of the recursion construct is that unfolding preserves the semantics. Since the actions of $rec\, P.\ p$ and $p[rec\, P.\ p/P]$ are identical this result is straightforward, but it does presuppose that R is reflexive.

Proposition 3.15 *If R is reflexive then $rec\, P.\ p \ \mathcal{B}^c(R) \ p[rec\, P.\ p/P]$.*

Finally we show that recursion induction is sound for our semantics when applied to guarded and sequential recursive definitions. Recall from [Mil89] that P is *guarded* in a term t if every occurrence of P in t appears within a guarded subterm, i.e. one of the form $a.t'$ or $< a \text{ at } \sigma x >. t'$. Also, P is *sequential* in t if every subterm of t which contains P, apart from P itself, is guarded or has the form $t_1 + t_2$.

Proposition 3.16 *If R is reflexive and transitive, P is guarded and sequential in t and t contains at most the variable P free, then $s \ \mathcal{B}^c(R) \ t[s/P]$ implies $s \ \mathcal{B}^c(R) \ rec\, P.\ t$.*

Proof. The proof is based on that of Proposition 7.13 of [Mil89], so we only give an outline here. Let p, q be any two terms such that $p \ \mathcal{B}^c(R) \ t[p/P]$ and $q \ \mathcal{B}^c(R) \ t[q/P]$ and let

$$G = \{ (t'[p/P], t'[q/P]) \mid P \text{ is sequential in } t' \}$$

Then one can show, for any $(t_0, t_1) \in G$, that

1. $t_0 \xrightarrow{\tau} t_0'$ implies $\exists\, t_1', t_1''$ such that $t_1 \stackrel{\tau}{\Rightarrow} t_1'$ and $t_0' \ \mathcal{B}(R) \ t_1'' \, G \, t_1'$

2. $t_0 \xrightarrow[u]{a} t_0'$ implies $\exists\, t_1'$ and t_1'' such that $t_1 \overset{a}{\underset{v}{\Longrightarrow}} t_1'$ and $t_0' \ \mathcal{B}(R) \ t_1'' \, G \, t_1'$ and $u \, R \, v$

3. similarly with t_0 and t_1 interchanged.

As in Proposition 3.14 this is sufficient to establish that if $(t_0, t_1) \in G$ then $t_0 \ \mathcal{B}^c(R) \ t_1$. Taking p to be s, t' to be t and q to be $rec\, P.\ t$ it follows that $s \ \mathcal{B}^c(R) \ rec\, P.\ t$. □

4 Axiomatisation

We have just seen that some interesting features of parameterized location bisimulations $\mathcal{B}(R)$ depend on specific properties of the underlying relation on locations R, as for example reflexivity, transitivity, and compatibility with concatenation. In this section we propose an axiomatisation, over the set of finite terms of $\mathbb{L}$, for parameterized location bisimulations $\mathcal{B}(R)$ based on particular relations R that we call *sensible*, or more accurately for their substitutive closure $\mathcal{B}^c(R)$. Formally:

Definition 4.1 A relation R on locations is called *sensible* if and only if it is of the form $R = \{\, (ul, vl) \,|\, u\,\widehat{R}\,v,\ l \in Loc \,\}$ for some relation on locations $\widehat{R}$ satisfying:

1. $\widehat{R}$ is a preorder
2. $\widehat{R}$ is compatible with concatenation on the left and on the right:

$$u\,\widehat{R}\,v \;\Rightarrow\; wu\,\widehat{R}\,wv \text{ and } uw\,\widehat{R}\,vw$$

3. $\widehat{R}$ is compatible with location renaming:

$$u\,\widehat{R}\,v \;\Rightarrow\; \pi(u)\,\widehat{R}\,\pi(v) \text{ for any } \pi : Loc \to Loc^*$$

□

Let us briefly comment on this definition. The prerequisite that R should be of the form $\{\, (ul, vl) \,|\, u\widehat{R}v,\ l \in Loc \,\}$ essentially translates into a requirement for the resulting *plb* $\mathcal{B}(R)$, namely that R-bisimilar processes should mark corresponding actions with the same location name l. This requirement will be used in our axiomatization. However, it is not strictly necessary for defining meaningful *plb*'s: for instance it is possible to show that the location equivalence $\approx_\ell$ defined as $\mathcal{B}(Id)$ on our language could also be obtained as the *plb* induced by the relation $R = \{(ul, uk) \,|\, u \in Loc^*,\ l, k \in Loc\}$, which is not of the required form. On the other hand, we will see that the requirement that R relates pairs of locations ending with the same letter is essential for defining the location preorder $\sqsubseteq_\ell$ in section 6.

Properties 1 and 2 for $\widehat{R}$ imply the same properties for R, and we saw in the previous section that such properties are natural if we want the resulting *plb* $\mathcal{B}(R)$ to be well-behaved. Property 3 expresses the fact that the particular choice of location names is irrelevant. Note that if $\widehat{R}$ satisfies property 3 then $\mathcal{B}(R)$ is compatible with location renaming, since the proof of Lemma 3.9 essentially refers to $\widehat{R}$'s (rather than R's) compatibility with location renaming.

Let us consider some examples. The identity relation *Id* on non-empty locations is obviously sensible, and is the strongest sensible relation. On the other hand (the inverses of) the *suffix* and the *prefix* relations, discussed in the previous section, are not compatible with concatenation, resp. on the left and on the right, and therefore are not sensible relations (nor may be used as the $\widehat{R}$ generating sensible relations). As another example, the relation U_ℓ given by $u\, U_\ell\, v \;\Leftrightarrow\; \exists l \in Loc\; \exists u', v'.\; u = u'l \;\&\; v = v'l$ is also a sensible relation, and in fact the weakest one. One can show that for processes of $\mathbb{P}$ the equivalence $\mathcal{B}(U_\ell)$ coincides with $\mathcal{B}(U)$ and thus with the bisimulation equivalence $\approx$:

Lemma 4.2 *For all processes p, q:* $\quad (p,q) \in \mathcal{B}(U_\ell)$ *if and only if* $p \approx q$.

Proof. The "only if" part results from Proposition 3.5 and Property 2.4. For the "if" part we show that $\approx$ is an U_ℓ-location bisimulation. If $p \approx q$ and $p \overset{a}{\underset{ul}{\Longrightarrow}} p'$ then there exist v, k and q' such that $q \overset{a}{\underset{vk}{\Longrightarrow}} q'$ and $p' \approx q'$. Let $h \notin loc(p) \cup loc(q)$. Then by Property 3.1 there exists q'' such that $q \overset{a}{\underset{vl}{\Longrightarrow}} q''[h \to l]$, and $q' = q''[h \to k]$. To conclude it is enough to note that $q' \approx q''[h \to l]$. $\square$

Since a sensible relation R is a preorder, the corresponding parameterized location bisimulation $\mathcal{B}(R)$ is also a preorder: it will then be denoted by $\sqsubseteq_R$, and the associated *precongruence* by $\sqsubseteq^c_R$. However, we shall maintain the notation $\approx_\ell$ for $\mathcal{B}(Id)$. In this section we propose an axiomatisation for any parameterized location precongruence $\sqsubseteq^c_R$ based on a sensible relation R, over the set $\mathbb{L}_f$ of finite terms of $\mathbb{L}$, i.e. terms built without process variables and recursion. We should point out however that our axiomatisation also holds for slightly less restricted relations R, where Property 2 is replaced simply by compatibility with concatenation on the left.

We show now a property which will be used in the axiomatisation, namely that a relation $\sqsubseteq^c_R$ induced by a sensible relation R treats free location variables essentially as fresh location names:

Lemma 4.3 (*Generalisation lemma*) *Let R be a sensible relation, and p, q be two terms with $Lvar(p) \cup Lvar(q) \subseteq \{x_1, \ldots, x_n\}$. Let $k_1, \ldots, k_n$ be distinct location names not occurring in p and q. Then:*

$$p[\,k_1/x_1, \ldots, k_n/x_n\,] \sqsubseteq_R q[\,k_1/x_1, \ldots, k_n/x_n\,] \;\Leftrightarrow\; p \sqsubseteq_R q$$

Proof. In one direction, namely "$\Leftarrow$", this is obvious. Conversely, let G be the relation on closed terms given by: $p[\rho]\, G\, q[\rho]$ if for some $k_1, \ldots, k_n$ satisfying the hypothesis of the lemma one has $p[k_1/x_1,\ldots,k_n/x_n] \sqsubseteq\!\!\!\!\sim_R q[k_1/x_1,\ldots,k_n/x_n]$. We prove that G is an R-location bisimulation. For this proof we shall abbreviate $r[k_1/x_1,\ldots,k_n/x_n]$ to $r[\vec{k}/\vec{x}]$. If $p[\rho] \overset{a}{\underset{ul}{\Longrightarrow}} p'$ then by lemma 3.4 there exist p'' and σ such that $u = \sigma[\rho]$, $p' = p''[\rho]$, and $p[\vec{k}/\vec{x}] \xrightarrow[\sigma[\vec{k}/\vec{x}]l]{a} p''[\vec{k}/\vec{x}]$. Then $q[\vec{k}/\vec{x}] \overset{a}{\underset{vl}{\Longrightarrow}} q'$ for some v and q' such that $\sigma[\vec{k}/\vec{x}]\,\widehat{R}\,v$ and $p''[\vec{k}/\vec{x}] \sqsubseteq\!\!\!\!\sim_R q'$. By lemma 3.4 again, there exist q'' and σ' such that $v = \sigma'[\vec{k}/\vec{x}]$, $q' = q''[\vec{k}/\vec{x}]$, and $q[\rho] \xRightarrow[\sigma'[\rho]l]{a} q''[\rho]$. Since the k_i's are distinct and do not occur in p and q, they will not occur in σ and σ' either, therefore we have $\sigma[\rho]\,\widehat{R}\,\sigma'[\rho]$ because $\widehat{R}$ is compatible with location renaming. Moreover $p''[\rho]\,G\,q''[\rho]$. Similarly a move $p[\rho] \overset{\varepsilon}{\Rightarrow} p'$ is matched by a move of $q[\rho]$. This shows that G is an R-location bisimulation. □

For the rest of this section we shall only consider sensible relations $R \subseteq Loc^* \times Loc^*$. Our proof system for $p \sqsubseteq\!\!\!\!\sim^c_R q$ will consist of a set of inequations of the form $p \sqsubseteq q$, together with some inference rules. We will use implicitly some standard axioms and inference rules, namely the ones expressing reflexivity, transitivity, and compatibility with the constructs. Also, we shall use equations $p = q$ to stand for the pair of inequations $p \sqsubseteq q$ and $q \sqsubseteq p$. For terms involving location variables, we have an inference rule corresponding to the generalisation lemma:

S1. If $Lvar(p) \cup Lvar(q) \subseteq \{x_1,\ldots,x_n\}$ and $k_1,\ldots,k_n$ are distinct location names not occurring in p and q then:

$$p[k_1/x_1,\ldots,k_n/x_n] \sqsubseteq q[k_1/x_1,\ldots,k_n/x_n] \quad \Rightarrow \quad p \sqsubseteq q$$

The first step of the axiomatisation consists as usual in reducing processes to *normal forms*, which are essentially notations for the transition systems used in the operational semantics. Here the normal forms will be terms built with $+$ and the prefixing construct $< a \text{ at } \sigma x >.p$. They are in fact a special kind of *head normal form*. More precisely:

Definition 4.4 A *head normal form* is a term (defined modulo axioms A1, A2, A3, see Fig. 4) of the form:

$$p = \sum_{i \in I} < a_i \text{ at } \sigma_i x_i >.p_i \; + \; \sum_{j \in J} \tau.p'_j$$

By convention this head normal form is *nil* if $I = \emptyset = J$. A *normal form* is a head normal form whose subterms are again normal forms. □

We introduce now the axioms that will allow us to transform terms of IL_f into normal forms. From now on, the laws will be given for closed terms; by virtue of S1, these laws can then be turned into similar statements on open terms. The basic transformation, replacing ordinary prefixing by the new prefixing construct, is the following:

L1. $\quad a.p \;=\; < a \text{ at } x >. x :: p$

We recall that the meaning of $< a \text{ at } x >. q$ is that action a occurs at some location l, instantiation of x, giving rise to the process q where x is replaced by l. Note that since p is closed, no variable capture may occur in applying L1.

The law L1 introduces a new location variable in front of the subterm p. We give now a set of laws to push locations through subterms. In particular we will have an axiom, L2, which will allow us to remove a location u on top of the prefixing construct
$< a \text{ at } vx >. q$, incrementing by u the access path v for the action a.

L2. $\quad u :: < a \text{ at } vx >. p \;=\; < a \text{ at } uvx >. u :: p$

L3. $\quad u :: \tau . p \;=\; \tau \,.\, u :: p$

L4. $\quad u :: nil \;=\; nil$

L5. $\quad u :: (p + q) \;=\; u :: p \;+\; u :: q$

Using laws L2, S1, we may infer for instance $y :: < a \text{ at } \sigma x >. p \;=\; < a \text{ at } y \sigma x >. y :: p$. Note however that this only holds for $y \neq x$, since the variable x is bound in $< a \text{ at } \sigma x >. p$. Indeed we need a kind of α-conversion rule:

S2. $\quad < a \text{ at } ux >.\, p \;=\; < a \text{ at } uy >.\, p[y/x] \;, \qquad y$ not free in p

Note that α-conversion is also involved in the substitution operation, which occurs for instance in applying rule S1. Let us now see an example of application of the laws L1 and L2 – where we also use implicitly some congruence laws. For the process $a.b.p$ we obtain:

$$a.b.p = < a \text{ at } x >. x :: < b \text{ at } y >. y :: p \;=\; < a \text{ at } x >. < b \text{ at } x\, y >. x :: y :: p$$

So far we have seen how prefixing $< a \text{ at } \sigma x >. p$ and location variables are introduced. In order to obtain normal forms, we also need to get rid of the static operators occurring in terms. The idea is as usual to eliminate the parallel operator by means of an *expansion theorem* (while the other static operators will be taken care of by standard laws, listed as R1–R4, U1–U4 in Fig. 4). In our case the expansion theorem will be as follows, where we use the notation $p[x\surd]$ introduced previously:

Expansion theorem: Let p, q be closed head normal forms:

$$p = \sum_{i \in I} < a_i \text{ at } u_i x_i >. p_i + \sum_{j \in J} \tau . p'_j \quad \text{and} \quad q = \sum_{k \in K} < b_k \text{ at } v_k y_k >. q_k + \sum_{l \in L} \tau . q'_l$$

Then the following law is sound for any parameterized location precongruence:

$$\begin{array}{lrl} \text{(EXP)} & p \mid q = & \sum_{i \in I} < a_i \text{ at } u_i x_i >. (p_i \mid q) + \sum_{k \in K} < b_k \text{ at } v_k y_k >. (p \mid q_k) + \\ & & \sum_{\bar{a}_i = b_k} \tau . (\, p_i [\, x_i \surd] \mid q_k [\, y_k \surd]\,) + \sum_{j \in J} \tau . (p'_j \mid q) + \sum_{l \in L} \tau . (p \mid q'_l) \end{array}$$

Note: The proof of soundness is given below, as part of the proof of Proposition 4.5.

Consider now the sets of equations $\mathcal{E}$ and $\mathcal{L}$ in figures 4 and 5. The equations $\mathcal{E}$ are more or less the standard expansion laws, adapted to account for the new prefixing construct. Together with the laws $\mathcal{L}$, which express properties of locations, these equations are used to reduce terms of $\mathbb{L}_f$ to (essentially) location transition systems. For instance these equations allow one to reduce the process $(l :: \alpha \mid \bar{\alpha} b) \backslash \alpha$ to $\tau . < b \text{ at } x >. nil$, while $(l :: (\alpha + b) \mid \bar{\alpha} b) \backslash \alpha$ can be shown equal to $< b \text{ at } ly >. nil + \tau . < b \text{ at } x >. nil$.

Similarly, the laws $\mathcal{T}$ in figure 6 are an adaptation of Milner's τ-laws to our language. To deal with the particular relation R on which the parameterized location precongruence $\sqsubseteq\!\!\!\sim^{c}_{R}$ is based, we have in addition a parametric inequation GEN_R; note that this is the only place where R intervenes in the axiomatisation. This inequation may be seen as an *absorption* law, expressing the fact that a location word may be subsumed by another one in the relation R, where q is said to be absorbed by p w.r.t. a relation G if $(p + q)\, G\, p$. Then an equivalent formulation of the axiom GEN_R is, with the hypothesis $u \,\widehat{R}\, v$:

$$< a \text{ at } ux >. p \sqsubseteq < a \text{ at } ux >. p + < a \text{ at } vx >. p \sqsubseteq < a \text{ at } vx >. p$$

Let now $\mathcal{I}_R$ be the set of all the laws and rules considered so far, including S1, S2. We write $p \sqsubseteq_R q$ if $p \sqsubseteq q$ is provable in this proof system, and similarly for $p =_R q$. We want to show that on terms of $\mathbb{L}_f$ the parameterized location precongruence $\sqsubseteq\!\!\!\sim^{c}_{R}$ coincides with $\sqsubseteq_R$. We start by proving that the laws $\mathcal{I}_R$ are sound for $\sqsubseteq\!\!\!\sim^{c}_{R}$.

$$
\begin{array}{lrcl}
\text{(A1)} & p+(q+r) & = & (p+q)+r \\
\text{(A2)} & p+q & = & q+p \\
\text{(A3)} & p+nil & = & p \\
\text{(A4)} & p+p & = & p
\end{array}
$$

$$
\begin{array}{lrcl}
\text{(R1)} & nil\backslash a & = & nil \\
\text{(R2)} & (<a \text{ at } ux>.p)\backslash b & = & \begin{cases} <a \text{ at } ux>.\,(p\backslash b) & \text{if } a \neq b, \bar{b} \\ nil & \text{otherwise} \end{cases} \\
\text{(R3)} & (\tau\,.\,p)\backslash b & = & \tau\,.\,(p\backslash b) \\
\text{(R4)} & (p\,+\,q)\backslash a & = & p\backslash a\,+\,q\backslash a
\end{array}
$$

$$
\begin{array}{lrcl}
\text{(U1)} & nil\,[f] & = & nil \\
\text{(U2)} & (<a \text{ at } ux>.p)\,[f] & = & <f(a) \text{ at } ux>.p\,[f] \\
\text{(U3)} & (\tau\,.p)\,[f] & = & \tau\,.(p\,[f]) \\
\text{(U4)} & (p\,+\,q)\,[f] & = & p[f]\,+\,q\,[f]
\end{array}
$$

(EXP) Let $p = \sum_{i\in I} <a_i \text{ at } u_i x_i>.p_i \;+\; \sum_{j\in J} \tau.p'_j$ and $q = \sum_{k\in K} <b_k \text{ at } v_k y_k>.q_k \;+\; \sum_{l\in L} \tau.q'_l$

Then:

$$
\begin{aligned}
p \mid q \;=\; & \sum_{i\in I} <a_i \text{ at } u_i x_i>.(p_i \mid q) \;+\; \sum_{k\in K} <b_k \text{ at } v_k y_k>.(p \mid q_k) \;+ \\
& \sum_{\bar{a}_i = b_k} \tau\,.(\,p_i\,[\,x_i\surd] \mid q_k\,[\,y_k\surd]\,) \;+\; \sum_{j\in J} \tau\,.(p'_j \mid q) \;+\; \sum_{l\in L} \tau\,.(p \mid q'_l)
\end{aligned}
$$

Figure 4: Equations $\mathcal{E}$, standard expansion laws.

$$
\begin{array}{lrcl}
\text{(L1)} & a.p & = & <a \text{ at } x>.\,x :: p \\
\text{(L2)} & u :: <a \text{ at } vx>.p & = & <a \text{ at } uvx>.u :: p \\
\text{(L3)} & u :: \tau.p & = & \tau.u :: p \\
\text{(L4)} & u :: nil & = & nil \\
\text{(L5)} & u :: (p+q) & = & u :: p \;+\; u :: q
\end{array}
$$

Figure 5: Equations $\mathcal{L}$, location laws.

$$
\begin{array}{llcl}
\text{(T1)} & p+\tau.p & = & \tau.p \\
\text{(T2)} & <a \text{ at } ux>.\, p & = & <a \text{ at } ux>.\, \tau.p \\
\text{(T2)}' & \tau.p & = & \tau.\tau.p \\
\text{(T3)} & <a \text{ at } ux>.\, (p+\tau.q) & = & <a \text{ at } ux>.\, (p+\tau.q) \; + <a \text{ at } ux>.\, q
\end{array}
$$

Figure 6: Equations $\mathcal{T}$, the τ-laws.

$$(\mathrm{GEN}_R) \quad \text{If} \quad (u,v) \in \widehat{R} \quad \text{then:} \quad <a \text{ at } ux>.\, p \;\sqsubseteq\; <a \text{ at } vx>.\, p$$

Figure 7: GEN_R, the parametric law.

Proposition 4.5 (*Soundness of the laws*) *The laws $\mathcal{I}_R$ are sound for the parameterized location precongruence $\sqsubseteq\!\!\!\!\sim^{c}_{R}$, that is $p \sqsubseteq_R q \;\Rightarrow\; p \sqsubseteq\!\!\!\!\sim^{c}_{R} q$.*

Proof. The implicit axioms and rules (for reflexivity, transitivity, and so on) are clearly sound. For the τ-laws, the proof of soundness is the usual one. The soundness of S1 is shown by taking terms $p+a$ and $q+a$ in the Generalisation Lemma 4.3. For the other laws it is enough to show that they are sound with respect to the *strong* version of $\sqsubseteq\!\!\!\!\sim^{c}_{R}$ (which is a precongruence satisfying the generalisation lemma 4.3 for any sensible relation R), defined in terms of "strong" arrows $p \xrightarrow[ul]{a} p'$ and $p \xrightarrow{\tau} p'$. We omit the proof regarding the standard equations (as well as S1); also the soundness of GEN_R is very easy to check. So we only prove here the soundness of the expansion theorem. Let G be the relation consisting of the identity pairs (s,s) and the pairs $((p \mid q), r)$ where:

$$p = \sum_{i\in I} <a_i \text{ at } u_i x_i>.\, p_i \;+\; \sum_{j\in J} \tau.p'_j \quad \text{and} \quad q = \sum_{k\in K} <b_k \text{ at } v_k y_k>.\, q_k \;+\; \sum_{l\in L} \tau.q'_l$$

and

$$
\begin{aligned}
r \;=\;& \sum_{i\in I} <a_i \text{ at } u_i x_i>.(p_i \mid q) \;+\; \sum_{k\in K} <b_k \text{ at } v_k y_k>.(p \mid q_k) \;+ \\
& \sum_{\bar{a}_i = b_k} \tau.(\,p_i[\,x_i\surd] \mid q_k[\,y_k\surd]) \;+\; \sum_{j\in J} \tau.(p'_j \mid q) \;+\; \sum_{l\in L} \tau.(p \mid q'_l)
\end{aligned}
$$

We show that G is a (strong) R-location bisimulation, for any reflexive relation R on locations. If $(p \mid q) \xrightarrow[u_i l]{a_i} (p_i[l/x_i] \mid q)$ then $r \xrightarrow[u_i l]{a_i} (p_i \mid q)[l/x_i]$ is the matching move of r, since q is closed and thus $(p_i \mid q)[l/x_i] = (p_i[l/x_i] \mid q)$. If $(p \mid q) \xrightarrow{\tau} (p_i[\,x_i\surd] \mid q_k[\,y_k\surd])$ with $\bar{a}_i = b_k$ by the transition law ST2, we obviously have the corresponding transition $r \xrightarrow{\tau} (p_i[\,x_i\surd] \mid q_k[\,y_k\surd])$ for

r. The remaining cases are obvious. $\square$

Let us turn now to the proof of completeness, that is $p \sqsubseteq\!\!\!\!\sim^{c}_{R} q \Rightarrow p \sqsubseteq_R q$. We show first that any term of $\mathbb{L}_f$ may be transformed into a head normal form, and then into a normal form, using the laws $\mathcal{E}$ and $\mathcal{L}$ (indeed the reduction to normal forms is independent of the choice of the relation R). To prove the normalisation result, we will use the notion of *norm* of a term p, noted $\|p\|$, defined as follows:

$$
\begin{aligned}
&\|nil\| = 0\\
&\|a.p\| = \|<a \text{ at } \sigma x>.p\| = \|\tau.p\| = 1 + \|p\|\\
&\|p \mid q\| = \|p\| + \|q\|\\
&\|p + q\| = max\{\|p\|, \|q\|\}\\
&\|p\backslash b\| = \|p\|\\
&\|p[f]\| = \|p\|\\
&\|\sigma :: p\| = \|p\|
\end{aligned}
$$

Thus $\|p\|$ is an upper bound on the maximal length of a transition sequence of p. Moreover, it can easily be shown that if $p \xrightarrow{\tau} p'$ or $p \xrightarrow[u]{a} p'$ then $\|p'\| < \|p\|$. We are now ready to prove the following:

Lemma 4.6 (*Head normalisation*) *For each term p of $\mathbb{L}_f$, there exists a head normal form* $\text{hnf}(p)$ *such that* $p =_R \text{hnf}(p)$ *and* $\|\text{hnf}(p)\| \leq \|p\|$.

Proof. First we show that it is enough to prove the statement for p closed. Assume that this has been done, and let p be an open term. Let $y_1, \ldots, y_n$ be the location variables occurring free in p, and let $k_1, \ldots, k_n$ be distinct location names, not occurring in p. We write $r[\vec{k}/\vec{y}]$ for $r[k_1/y_1, \ldots, k_n/y_n]$. Then $p[\vec{k}/\vec{y}] =_R q$ for some (closed) head normal form q. By α-conversion we may assume that the y_i's are not bound in q. We let $\text{hnf}(p)$ be the term we obtain from q by replacing the k_i's by the y_i's. Clearly $\text{hnf}(p)$ is a head normal form, where the k_i's do not occur, and $q = \text{hnf}(p)[\vec{k}/\vec{y}]$. Therefore $p =_R \text{hnf}(p)$ by S1. Moreover it should be obvious that $\|\text{hnf}(p)\| \leq \|p\|$ since $\|\text{hnf}(p)\| = \|q\| \leq \|p[\vec{k}/\vec{y}]\| = \|p\|$.

We prove now the lemma for closed terms, by structural induction (one could check that we do not need the induction hypothesis on open subterms). The proof makes use of all axioms in $\mathcal{E}$ and $\mathcal{L}$ – except for the idempotence law A4. As usual, we will use axioms A1, A2, A3 without mentioning them.

1) for $p = nil$ we let $\text{hnf}(p) = nil$.

2) $p = a.q$. We define $\mathrm{hnf}(p) = < a \text{ at } x >. x :: q$. Then we have $p =_R \mathrm{hnf}(p)$ by law L1, and $\|\mathrm{hnf}(p)\| = 1 + \|x :: q\| = 1 + \|q\| = \|p\|$.

3) for $p = \tau.q$ or $p = < a \text{ at } ux >.q$ we let $\mathrm{hnf}(p) = p$.

4) $p = u :: q$. By induction there exists $\mathrm{hnf}(q)$ such that $q =_R \mathrm{hnf}(q)$ and $\|\mathrm{hnf}(q)\| \leq \|q\|$. Now if $\mathrm{hnf}(q) = nil$, we let $\mathrm{hnf}(p) = nil$. Then we have $p =_R nil$ by L4. Otherwise let $\mathrm{hnf}(q) = \sum_{i \in I} < a_i \text{ at } v_i x_i >. q_i + \sum_{j \in J} \tau . q'_j$. We define now

$$\mathrm{hnf}(p) = \sum_{i \in I} < a_i \text{ at } uv_i x_i >. u :: q_i + \sum_{j \in J} \tau . u :: q'_j$$

Then using laws L2, L3, L5 we obtain

$$\begin{aligned} p &=_R \sum_{i \in I} u :: < a_i \text{ at } v_i x_i >. q_i + \sum_{j \in J} u :: \tau . q'_j \\ &=_R \sum_{i \in I} < a_i \text{ at } uv_i x_i >. u :: q_i + \sum_{j \in J} \tau . u :: q'_j = \mathrm{hnf}(p) \end{aligned}$$

Since $\|p\| = \|q\|$ by definition, and it is easy to see that $\|\mathrm{hnf}(p)\| = \|\mathrm{hnf}(q)\|$, we may conclude that $\|\mathrm{hnf}(p)\| \leq \|p\|$.

5) $p = r \mid q$. By induction there exist $\mathrm{hnf}(r)$, $\mathrm{hnf}(q)$, such that $r =_R \mathrm{hnf}(r)$ and $q =_R \mathrm{hnf}(q)$, with $\|\mathrm{hnf}(r)\| \leq \|r\|$ and $\|\mathrm{hnf}(q)\| \leq \|q\|$. If $\mathrm{hnf}(r)$, $\mathrm{hnf}(q)$ are as follows:

$$\mathrm{hnf}(r) = \sum_{i \in I} < a_i \text{ at } u_i x_i >. r_i + \sum_{j \in J} \tau . r'_j, \quad \mathrm{hnf}(q) = \sum_{k \in K} < b_k \text{ at } v_k y_k >. q_k + \sum_{l \in L} \tau . q'_l$$

let:

$$\begin{aligned} \mathrm{hnf}(p) = \quad & \sum_{i \in I} < a_i \text{ at } u_i x_i >. (r_i \mid q) + \sum_{k \in K} < b_k \text{ at } v_k y_k >. (r \mid q_k) + \\ & \sum_{\bar{a}_i = b_k} \tau . (\, r_i [\, x_i \surd] \mid q_k [\, y_k \surd] \,) + \sum_{j \in J} \tau . (r'_j \mid q) + \sum_{l \in L} \tau . (r \mid q'_l) \end{aligned}$$

Then we have $p =_R \mathrm{hnf}(p)$ by induction and by the expansion theorem. Note now that

$$\begin{aligned} \|\mathrm{hnf}(p)\| &= 1 + max \{ \|r_i\| + \|q\|, \|r\| + \|q_k\|, \|r'_j\| + \|q\|, \|r\| + \|q'_l\| \} \\ &\leq \|r\| + \|q\| = \|p\| \end{aligned}$$

6) $p = r + q$. By induction r, q have head normal forms $\mathrm{hnf}(r)$, $\mathrm{hnf}(q)$, such that $r =_R \mathrm{hnf}(r)$ and $q =_R \mathrm{hnf}(q)$. Define $\mathrm{hnf}(p) = \mathrm{hnf}(r) + \mathrm{hnf}(q)$. Then $\mathrm{hnf}(p)$ is already a head normal form, since hnf's are defined modulo axioms A1, A2, A3, and $\|\mathrm{hnf}(p)\| \leq \|p\|$ follows easily by induction.

7) $p = q\backslash b$. By induction there exists $\mathrm{hnf}(q)$ such that $q =_R \mathrm{hnf}(q)$ and $\|\mathrm{hnf}(q)\| \leq \|q\|$. If $\mathrm{hnf}(q) =_R nil$ let $\mathrm{hnf}(p) = nil$. Then $p =_R \mathrm{hnf}(p)$ by law R1, and it is obvious that $\|\mathrm{hnf}(p)\| \leq \|p\|$. Otherwise, if $\mathrm{hnf}(q) = \sum_{i\in I} < a_i \text{ at } u_i x_i >. q_i + \sum_{j\in J} \tau . q'_j$, we let $\mathrm{hnf}(p) = \sum_{a_i \neq b, \bar{b}} < a_i \text{ at } u_i x_i >. (q_i \backslash b) + \sum_{j\in J} \tau . (q'_j \backslash b)$. Then we obtain $p =_R \mathrm{hnf}(p)$ by laws R2, R3, R4, and by induction $\|\mathrm{hnf}(p)\| \leq \|p\|$. In particular if there is an $i \in I$ such that $a_i = b, \bar{b}$ then $\|\mathrm{hnf}(p)\| < \|p\|$: this is the only case where the norm decreases.

8) The case $p = q[f]$ is similar to 7). It makes use of laws U1,U2,U3,U4. □

Proposition 4.7 (*Normalisation*) *For each term p of $\mathbb{L}_f$, there exists a normal form* $\mathrm{nf}(p)$ *such that* $p =_R \mathrm{nf}(p)$ *and* $\|\mathrm{nf}(p)\| \leq \|p\|$.

Proof. The proof uses the previous lemma, and proceeds by induction on the norm $\|p\|$. If $\|p\| = 0$, we have $\mathrm{hnf}(p) = nil$ since $\|\mathrm{hnf}(p)\| \leq \|p\|$. In this case we let $\mathrm{nf}(p) = nil$. Otherwise, either $\mathrm{hnf}(p) = nil$ (e.g. if $p = a.nil\backslash a$), in which case we let $\mathrm{nf}(p) = nil$, or $\mathrm{hnf}(p) = \sum_{i\in I} < a_i \text{ at } \sigma_i x_i >. p_i + \sum_{j\in J} \tau . p'_j$. Here $\|p_i\| < \|\mathrm{hnf}(p)\| \leq \|p\|$ for any $i \in I$, and similarly for the p'_j's, therefore by induction these subterms have normal forms $\mathrm{nf}(p_i)$ and $\mathrm{nf}(p'_j)$. We let then $\mathrm{nf}(p) = \sum_{i\in I} < a_i \text{ at } \sigma_i x_i >. \mathrm{nf}(p_i) + \sum_{j\in J} \tau . \mathrm{nf}(p'_j)$.

Clearly the norm cannot increase during the normalisation process, since normalisation is nothing but recursive head normalisation. □

The proof of completeness requires in addition two *absorption lemmas*, similar to those used for weak bisimulation in [HM85].

Lemma 4.8 (*τ-absorption lemma*) *If p is a closed normal form then:*

$$p \stackrel{\tau}{\Rightarrow} p' \quad \textit{implies} \quad p + \tau . p' =_R p$$

Proof. By induction on the length of $p \stackrel{\tau}{\Rightarrow} p'$. The proof uses axioms A4 and T1. If $p = \sum_{i\in I} < a_i \text{ at } u_i x_i >. p_i + \sum_{j\in J} \tau . p'_j$, then the transition $p \stackrel{\tau}{\Rightarrow} p'$ is due to the part $\sum_{j\in J} \tau . p'_j$ of p. Now there are two possibilities:

i) $p'_j = p'$, for some $j \in J$. Then $p =_R p + \tau . p'$ by A4.

ii) $p'_j \overset{\tau}{\Rightarrow} p'$, for some $j \in J$. By induction $p'_j =_R p'_j + \tau . p'$. Then using axiom T1 we obtain: $p =_R p + \tau . p'_j =_R p + \tau . p'_j + p'_j =_R p + \tau . p'_j + p'_j + \tau . p' =_R p + \tau . p'$.

□

Lemma 4.9 (*General absorption lemma*) *If* p *is a closed normal form then:*

$$p \overset{a}{\underset{ul}{\Rightarrow}} p' \quad \textit{implies} \quad \exists p''.\ p =_R p + < a \text{ at } ux >.\ p'' \textit{ and } p''[l/x] = p'$$

Proof. By induction on the length of $p \overset{a}{\underset{ul}{\Rightarrow}} p'$ (or more precisely, on the number of $\tau's$ preceding the observable action). The proof uses axioms A4, T1, T3, as well as the above τ-absorption lemma. If $p = \sum_{i \in I} < a_i \text{ at } u_i x_i >. p_i + \sum_{j \in J} \tau . p'_j$, there are two possibilities for $p \overset{a}{\underset{ul}{\Rightarrow}} p'$.

i) $p \overset{a}{\underset{ul}{\Rightarrow}} p'$ because $a = a_i$, $u = u_i$ and $< a_i \text{ at } u_i x_i >. p_i \overset{a}{\underset{ul}{\Rightarrow}} p_i[l/x_i] \overset{\varepsilon}{\Rightarrow} p'$ for some $i \in I$. Now if $p' = p_i[l/x_i]$, we take $p'' = p_i$ and we get $p =_R p + < a_i \text{ at } u_i x_i >. p''$ using law A4. Otherwise we have $p_i[l/x_i] \overset{\tau}{\Rightarrow} p'$. Now by Lemma 3.4 there exists p'' s.t. $p' = p''[l/x_i]$ and $\forall k.\ p_i[k/x_i] \overset{\tau}{\Rightarrow} p''[k/x_i]$. Then we have $p_i[k/x_i] =_R p_i[k/x_i] + \tau . p''[k/x_i]$ by the τ-absorption lemma. We may thus apply S1 to infer $p_i =_R p_i + \tau . p''$. We now deduce, using A4, T1, T3:

$$\begin{aligned} p &=_R p + < a_i \text{ at } u_i x_i >. p_i \\ &=_R p + < a_i \text{ at } u_i x_i >. (p_i + \tau . p'') \\ &=_R p + < a_i \text{ at } u_i x_i >. (p_i + \tau . p'') + < a_i \text{ at } u_i x_i >. p'' \\ &=_R p + < a_i \text{ at } u_i x_i >. p'' \end{aligned}$$

ii) Otherwise $p \overset{a}{\underset{ul}{\Rightarrow}} p'$ because for some $j \in J$ we have $p'_j \overset{a}{\underset{ul}{\Rightarrow}} p'$. Then we may apply induction to get $p'_j =_R p'_j + < a \text{ at } ux >. p''$ for some p'' s.t. $p''[l/x] = p'$. We now deduce, using T1, A4:

$$\begin{aligned} p &=_R p + \tau . p'_j \\ &=_R p + \tau . p'_j + p'_j + < a \text{ at } ux >. p'' \\ &=_R p + < a \text{ at } ux >. p'' \end{aligned}$$

□

We should point out here that it would be possible to show a similar absorption lemma for our semantics of [BCHK91] – where we allow the actual location l of an action (in rules LT1 and LT2) to be a word instead of an atomic location. However such a lemma would not be sufficient for establishing a completeness result. This is a technical justification for considering the different way of observing localities adopted here.

We may now establish the announced completeness result. In the proof we will use the following characterisation for $\lesssim_R$, an adaptation of a similar characterisation for weak bisimulation $\approx$:

$$p \lesssim_R q \;\Leftrightarrow\; \begin{cases} \quad p \lesssim_R^c q \\ or \;\; \tau.p \lesssim_R^c q \\ or \;\; p \lesssim_R^c \tau.q \end{cases}$$

Theorem 4.10 (*Completeness*) *For any terms $p, q \in \mathbb{L}_f$: $p \lesssim_R^c q \;\Rightarrow\; p \sqsubseteq_R q$.*

Proof. We show first that it is enough to prove the statement for closed terms. Let p, q be open terms with free variables $x_1, \ldots, x_n$, and let $k_1, \ldots, k_n$ be distinct location names not occurring in p and q. If $p \lesssim_R^c q$ then $p[\vec{k}/\vec{x}] \lesssim_R^c q[\vec{k}/\vec{x}]$ and thus, assuming that we have proved completeness for closed terms, we have $p[\vec{k}/\vec{x}] \sqsubseteq_R q[\vec{k}/\vec{x}]$, therefore $p \sqsubseteq_R q$ by S1.

We prove now the theorem for closed terms. By virtue of the normalisation lemma and the soundness of the axioms, it is enough to prove the result for normal forms p, q. We will use implicitly in the proof the fact that terms obtained by transitions from normal forms are again normal forms. We proceed by induction on the sum of norms of p and q. We show $p \sqsubseteq_R q$ by proving that $p \sqsubseteq_R p + q$ and $p + q \sqsubseteq_R q$. We start by proving $p + q \sqsubseteq_R q$. Suppose that $p = \sum_{i \in I} < a_i \text{ at } u_i x_i >. p_i \; + \; \sum_{j \in J} \tau.p'_j$. We prove separately:

i) $q + \tau.p'_j \sqsubseteq_R q \qquad \forall j \in J$

ii) $q + < a_i \text{ at } u_i x_i >. p_i \sqsubseteq_R q \qquad \forall i \in I$

Proof of i). We have $p \overset{\tau}{\Rightarrow} p'_j$. Correspondingly, since $p \lesssim_R^c q$, there exists q' s.t. $q \overset{\tau}{\Rightarrow} q'$ and $p'_j \lesssim_R q'$. We know that $\|p'_j\| < \|p\|$ and $\|q'\| < \|q\|$. There are now three cases to consider. Suppose first that $\tau.p'_j \lesssim_R^c q'$. Then by induction $\tau.p'_j \sqsubseteq_R q'$ (note that here it is necessary to use an induction on the *sum* of norms), and thus, prefixing both terms by τ and using axiom T2′, we obtain $\tau.p'_j \sqsubseteq_R \tau.q'$. We may now use the τ-absorption lemma to get: $q + \tau.p'_j \sqsubseteq_R q + \tau.q' =_R q$. For the cases $p'_j \lesssim_R^c q'$ and $p'_j \lesssim_R^c \tau.q'$ we proceed similarly.

Proof of ii). Let $l \notin loc(p) \cup loc(q)$. We have $p \overset{a_i}{\underset{u_i l}{\Longrightarrow}} p_i[l/x_i]$. Since $p \sqsubseteq_R q$, there must exist v, q' s.t. $q \overset{a_i}{\underset{vl}{\Longrightarrow}} q'$, with $u_i \widehat{R} v$ and $p_i[l/x_i] \sqsubseteq_R q'$. Again we know that $\|p_i[l/x_i]\| < \|p\|$ and $\|q'\| < \|q\|$. Now by the generalized absorption lemma there exists q'' such that $q' = q''[l/x_i]$ and $q =_R q + \langle a_i \text{ at } vx_i\rangle.\, q''$. Again there are three cases. Suppose $\tau.p_i[l/x_i] \sqsubseteq^c_R q''[l/x_i]$. Then by induction $\tau.p_i[l/x_i] \sqsubseteq_R q''[l/x_i]$. Since $l \notin loc(p) \cup loc(q)$ by S1 we have $\tau.p_i \sqsubseteq_R q''$. Then prefixing both terms by τ and applying T2′, we get $\tau.p_i \sqsubseteq_R \tau.q''$. We thus obtain, using T2 and the parametric law GEN$_R$:

$$
\begin{aligned}
q + \langle a_i \text{ at } u_i x_i\rangle.p_i &=_R q + \langle a_i \text{ at } vx_i\rangle.q'' + \langle a_i \text{ at } u_i x_i\rangle.p_i \\
&=_R q + \langle a_i \text{ at } vx_i\rangle.\tau.q'' + \langle a_i \text{ at } u_i x_i\rangle.\tau.p_i \\
&\sqsubseteq_R q + \langle a_i \text{ at } vx_i\rangle.\tau.q'' + \langle a_i \text{ at } u_i x_i\rangle.\tau.q'' \\
&\sqsubseteq_R q + \langle a_i \text{ at } vx_i\rangle.\tau.q'' =_R q
\end{aligned}
$$

For the cases $p_i[l/x_i] \sqsubseteq^c_R q'$ and $p_i[l/x_i] \sqsubseteq^c_R \tau.q'$ we proceed similarly. This ends the proof of $p + q \sqsubseteq_R q$. The proof of $p \sqsubseteq_R p + q$ is symmetric. □

This concludes our axiomatization for parameterized location bisimulations $\mathcal{B}(R)$ based on a sensible relation R. In the next sections we will examine two particular instances of *plb's* axiomatizable in this way, namely the location equivalence $\approx_\ell$ and the location preorder $\sqsubseteq_\ell$.

5 Location Equivalence

In this section we discuss the generalized location bisimulation $\mathcal{B}(R)$ obtained by instantiating R as the identity relation Id. We recall that this is an equivalence relation, called *location equivalence* and denoted $\approx_\ell$. Clearly the identity relation on locations is a sensible relation, therefore our axiomatization result of the previous section holds for $\approx_\ell$, or more accurately for the associated congruence $\approx^c_\ell$. Note that the parametric absorption law GEN$_R$ is trivial in this case.

We already saw that location equivalence is strictly finer than bisimulation equivalence $\approx$. The example we gave, namely $p = (a.\alpha.c \mid b.\bar{\alpha}.d)\backslash\alpha$ and $q = (a.\alpha.d \mid b.\bar{\alpha}.c)\backslash\alpha$, also shows that location equivalence is different from Darondeau and Degano's *(weak) causal bisimulation* [DD89,DD90]: denoting by $\approx_c$ the causal weak bisimulation, we have $p \approx_c q$ since, roughly speaking, both c and d causally depend on a and b in p and q. However $p \not\approx_\ell q$ since in p the d action is not spatially dependent upon the a action. We can also give examples not involving the restriction operator to show that the two equivalences $\approx_\ell$ and $\approx_c$ are incomparable: let

$r = (a.\alpha + b.\beta \mid \bar{\alpha}.b + \bar{\beta}.a)$. Then

$$\begin{array}{ccccc} r + (a \mid b) & \approx_\ell & r & \not\approx_\ell & r + a.b \\ & \not\approx_c & & \approx_c & \end{array}$$

Note also that

$$(a \mid b) \;\approx_\ell\; (r \backslash \alpha, \beta) \;\approx_c\; a.b + b.a$$

These absorption phenomena, resp. of $(a \mid b)$ in r w.r.t. $\approx_\ell$, and of $a.b$ in r w.r.t. $\approx_c$, clearly show the difference between the two equivalences: the former equates processes with the same parallel structure, while the latter equates processes with the same causal structure. However, for a language without communication (and restriction) the two equivalences coincide, because in this case causal dependency coincides with spatial dependency. This is formalised in [Kie91], where the two equivalences are characterised as instantiations of the same general transition system: the two instantiations are equal if the restricted language is considered. We refer to this paper for a precise study of the relation between causal and location equivalences.

The equivalence $\approx_\ell$ is very similar to what is also called "location equivalence" in [BCHK91]: in both cases, equivalent processes have to perform the same actions at the same locations. In fact all motivations and examples given there to justify the introduction of this equivalence apply to the definition given here as well. In the rest of this section we will show how the two location equivalences relate to each other, and also give a comparison with another equivalence based on spatial distribution of processes, *distributed bisimulation* (see [CH89], [Cas88], [Kie89]).

We start by comparing the location equivalence $\approx_\ell$ with that of [BCHK91]. While the definition of the two equivalences is formally the same, the underlying location transition systems are slightly different. Transitions in [BCHK91] are more general in that the location allocated at each step is a word $u \in Loc^*$ instead of an atomic location $l \in Loc$. To avoid confusion we will call the transitions of [BCHK91], adapted to our new language, *loose location transitions*, noted $\overset{a}{\underset{u}{\rightsquigarrow}}$. For this transition system the rule (LT1) of Figure 1 is replaced by

$$\text{(LLT1)} \quad a.p \overset{a}{\underset{u}{\rightsquigarrow}} u :: p \qquad u \in Loc^*$$

The rule LT2 for $< a \text{ at } ux >.p$ is relaxed in the same way. The rules concerning the other process constructors, the τ-transitions and weak transitions are the same for the two transition systems. We denote the weak loose transitions by $p \overset{a}{\underset{u}{\Rrightarrow}} p'$. The location equivalence based on loose transitions will be called here *loose location equivalence* and denoted $\approx_{\ell\ell}$. We choose this name because the loose transition system gives more freedom to relate the behaviours of processes. In the location equivalence $\approx_\ell$, based on atomic allocation (i.e. LT1), we implicitly require the equality of the last allocated locations, while this is not true for loose location equivalence. The latter can introduce more than one atomic location within one move and thus is able to fill up "missing locations". The following example shows that in this way loose location equivalence $\approx_{\ell\ell}$ can equate processes which are distinguished by location equivalence $\approx_{\ell}$. Let p and q represent respectively the processes $(l :: \alpha \mid \bar{\alpha}.b)\backslash\alpha$ and $(l :: (\alpha + b) \mid \bar{\alpha}.b)\backslash\alpha$. Then the move of q

$$(l :: (\alpha + b) \mid \bar{\alpha}.b)\backslash\alpha \xrightarrow[lk]{b} (l :: k :: nil \mid \bar{\alpha}.b)\backslash\alpha$$

which is also a loose move, can only be matched by a loose move of p, introducing the location lk in one step:

$$(l :: \alpha \mid \bar{\alpha}.b)\backslash\alpha \overset{b}{\underset{lk}{\Rrightarrow}} (l :: nil \mid lk :: nil)\backslash\alpha$$

Indeed we have $p \approx_{\ell\ell} q$ but $p \not\approx_{\ell} q$. This also shows that the CCS terms $(a.\alpha \mid \bar{\alpha}.b)\backslash\alpha$ and $(a.(\alpha + b) \mid \bar{\alpha}.b)\backslash\alpha$ are loosely location equivalent but not location equivalent, and intuitively we want to distinguish these two processes since in the first the b action is not spatially dependent upon the a action. From this example we see that the two location equivalences are different for CCS terms. However the property of filling up "missing locations" only comes into play when processes containing restrictions are considered. We shall show now that for finite restriction-free processes the two equivalences coincide.

We show first that location equivalence is finer than loose location equivalence. To this end we need a few auxiliary assertions on the underlying transition systems that we state here without proofs (which are easy).

Lemma 5.1 *For any term p:*

1) $p \overset{a}{\underset{ul}{\Longrightarrow}} p'$ *and* $l \notin loc(p)$ *implies* $p \overset{a}{\underset{uv}{\Rrightarrow}} p'[l \to v]$ *for any* $v \in Loc^*$

2) *Let L be such that* $loc(p) \subseteq L \subset Loc$. *Then* $p \overset{a}{\underset{u}{\Rrightarrow}} p'$ *implies* $\exists v \in loc(p)^* \, \exists w.\ u = vw$ *and* $\forall l \notin L\ \exists p''.\ p \overset{a}{\underset{vl}{\Longrightarrow}} p''$ *and* $p''[l \to w] = p'$

In particular, if $p \overset{a}{\underset{u}{\Rrightarrow}} p'$ for some u and p' then there exist $v \in loc(p)^*$ and p'' such that $u = vw$ and $p \overset{a}{\underset{v}{\Rrightarrow}} p''$.

Proposition 5.2 $p \approx_{\ell} q \;\Rightarrow\; p \approx_{\ell\ell} q$.

Proof. We show that $\approx_{\ell}$ is a loose location bisimulation, relative to the identity relation on Loc. Assume that $p \approx_{\ell} q$ and $p \overset{a}{\underset{u}{\Rrightarrow}} p'$, and let $l \notin loc(p) \cup loc(q)$. Then by the previous lemma there exist v and w such that $u = vw$ and $p \overset{a}{\underset{vl}{\Longrightarrow}} p''$ with $p' = p''[l \to w]$. Then there exists q'' such that $q \overset{a}{\underset{vl}{\Longrightarrow}} q''$ and $p'' \approx_{\ell} q''$, therefore by the previous lemma $q \overset{a}{\underset{u}{\Rrightarrow}} q''[l \to w]$, and $p''[l \to w] \approx_\ell q''[l \to w]$ by Corollary 3.10. $\square$

The converse of this proposition only holds for finite processes without restriction. To establish this result, we need a "decomposition lemma", analogous to the Proposition 3.15 of [BCHK91]. To this end, we define the *observable length* of a finite process p, denoted $|p|$, to be the maximal length of a chain of observable actions of p, that is:

$$\forall p \in \mathbb{P}: \quad |p| = max\ \{\, n \mid p \overset{a_1}{\Rightarrow} \cdots \overset{a_n}{\Rightarrow} p' \,\}$$

Lemma 5.3 *Let $(u :: p \mid r)$ and $(u :: q \mid s)$ be two finite processes satisfying*

(i) $(loc(p) \cup loc(q)) \cap (loc(r) \cup loc(s)) = \emptyset$

(ii) *if* $r \overset{a}{\underset{v}{\Longrightarrow}} r'$ *or* $s \overset{a}{\underset{v}{\Longrightarrow}} s'$ *then v is not a prefix of u.*

Then if $(u :: p \mid r) \approx_{\ell\ell} (u :: q \mid s)$ one has

(1) $|p| = |q|$ *and* $|r| = |s|$

(2) $p \approx_{\ell\ell} q$ *and* $r \approx_{\ell\ell} s$.

Proof. First we note that $|(u :: p \mid r)| = |(u :: q \mid s)|$ since $(u :: p \mid r) \approx (u :: q \mid s)$, therefore $|p| + |r| = |q| + |s|$ since $|u :: p| = |p|$ and $|(t \mid r)| = |t| + |r|$. Then to show the first point it is enough to show that $|p| = |q|$. Let $p \overset{a_1}{\Rightarrow} \cdots \overset{a_n}{\Rightarrow} p'$. It is easy to see that, due to the assumption (i), there exist $v_1, \ldots, v_n$ such that $p \overset{a_1}{\underset{v_1}{\Longrightarrow}} \cdots \overset{a_n}{\underset{v_n}{\Longrightarrow}} p''$ where v_i does not contain any location of s. Then one has

$$(u :: q \mid s) \overset{a_1}{\underset{uv_1}{\Longrightarrow}} \cdots \overset{a_n}{\underset{uv_n}{\Longrightarrow}} (u :: q' \mid s')$$

The component s cannot be responsible for any a_i-move, since otherwise there would exist i and a prefix v of uv_i such that $s \overset{a}{\underset{v}{\Longrightarrow}} s''$. But then, due to the hypothesis (ii), v_i would contain a location of $loc(s)$. This shows $|p| \leq |q|$, and a symmetric argument shows $|q| \leq |p|$.

To prove $p \approx_{\ell\ell} q$, let G be the relation given by: $\hat{p}\, G\, \hat{q}$ if and only if there exist u, p, q, r and s satisfying the hypotheses required for the point (1) above, and there exists a location renaming π such that $\hat{p} = \pi(p)$ and $\hat{q} = \pi(q)$. We show that G is a (loose) location bisimulation. If $\pi(p) \overset{a}{\underset{v}{\Longrightarrow}} p'$ then by the Lemma 3.5 of [BCHK91] there exist a location renaming π', a location v' and a term p'' such that $p \overset{a}{\underset{v'}{\Longrightarrow}} p''$ with $\pi'(p) = \pi(p)$, $\pi'(v') = v$ and $\pi'(p'') = p'$ (and also $\pi'(q) = \pi(q)$). Let $l \in Loc - (loc(r) \cup loc(s))$. Then by the Lemma 5.1 above there exist $w \in loc(p)^*$, $w' \in Loc^*$ and $\bar{p}$ such that $v' = ww'$ and $p \overset{a}{\underset{wl}{\Longrightarrow}} \bar{p}$ with $p'' = \bar{p}[l \to w']$. Since $loc(\bar{p}) \subseteq loc(p) \cup \{l\}$ one has $loc(\bar{p}) \cap (loc(r) \cup loc(s)) = \emptyset$. From $(u :: p \mid r) \overset{a}{\underset{uwl}{\Longrightarrow}} (u :: \bar{p} \mid r)$ and the hypothesis $(u :: p \mid r) \approx_{\ell\ell} (u :: q \mid s)$ we infer that there exist $\bar{q}$ and $\bar{s}$ such that $(u :: q \mid s) \overset{a}{\underset{uwl}{\Longrightarrow}} (u :: \bar{q} \mid \bar{s})$ and $(u :: \bar{q} \mid \bar{s}) \approx_{\ell\ell} (u :: \bar{p} \mid r)$. The term s cannot be responsible for the transition $\overset{a}{\underset{uwl}{\rightsquigarrow}}$, since this would contradict either (i) or (ii) or $l \notin loc(s)$. Then $loc(\bar{q}) \subseteq loc(q) \cup loc(wl)$ and $loc(\bar{s}) \subseteq loc(s)$, therefore $(loc(\bar{p}) \cup loc(\bar{q})) \cap (loc(r) \cup loc(\bar{s})) = \emptyset$. Moreover one can see that $\bar{s}$ satisfies (ii), since otherwise s would not satisfy (ii). Then the pair of terms $(u :: \bar{p} \mid r)$ and $(u :: \bar{q} \mid \bar{s})$ satisfies the hypotheses required for the point (1) above, hence $|\bar{p}| = |\bar{q}|$ and $|r| = |\bar{s}|$. From this we infer that in the transition $(u :: q \mid s) \overset{a}{\underset{uwl}{\Rrightarrow}} (u :: \bar{q} \mid \bar{s})$ there cannot be any communication between q and s, since otherwise we would have $|\bar{s}| < |s| = |r|$. We conclude that $u :: q \overset{a}{\underset{uwl}{\Rrightarrow}} u :: \bar{q}$ (and $s \overset{\varepsilon}{\Rightarrow} \bar{s}$), that is $q \overset{a}{\underset{wl}{\Rrightarrow}} \bar{q}$. Then by the Lemma 3.5 of [BCHK91] $\pi(q) = \pi'(q) \overset{a}{\underset{v}{\Rrightarrow}} \pi'(\bar{q}[l \to w'])$, and $p' \, G \, \pi'(\bar{q}[l \to w'])$.

By symmetry, if $\pi(q) \overset{a}{\underset{v}{\Rrightarrow}} q'$ then there exists p' such that $\pi(p) \overset{a}{\underset{v}{\Rrightarrow}} p'$ and $p' \, G \, q'$. Now assume that $\pi(p) \overset{\varepsilon}{\Rightarrow} p'$. Then by the Lemma 3.5 of [BCHK91] there exists $\bar{p}$ such that $p' = \pi(\bar{p})$ and $p \overset{\varepsilon}{\Rightarrow} \bar{p}$. Since $(u :: p \mid r) \overset{\varepsilon}{\Rightarrow} (u :: \bar{p} \mid r)$ there exist $\bar{q}$ and $\bar{s}$ such that $(u :: q \mid s) \overset{\varepsilon}{\Rightarrow} (u :: \bar{q} \mid \bar{s})$ and $(u :: \bar{p} \mid r) \approx_{\ell\ell} (u :: \bar{q} \mid \bar{s})$. It should be clear that the pair of terms $(u :: \bar{p} \mid r)$ and $(u :: \bar{q} \mid \bar{s})$ satisfies the hypotheses required for the point (1) above, hence $|\bar{p}| = |\bar{q}|$ and $|r| = |\bar{s}|$. Then in the transition $(u :: q \mid s) \overset{\varepsilon}{\Rightarrow} (u :: \bar{q} \mid \bar{s})$ there cannot be any communication between q and s, therefore $q \overset{\varepsilon}{\Rightarrow} \bar{q}$ (and $s \overset{\varepsilon}{\Rightarrow} \bar{s}$), and if we let $q' = \pi(\bar{q})$ we have $p' \, G \, q'$ and $\pi(q) \overset{\varepsilon}{\Rightarrow} q'$ (Lemma 3.5 of [BCHK91]).

Clearly, by symmetry if $\pi(q) \overset{\varepsilon}{\Rightarrow} q'$ then there exists p' such that $\pi(p) \overset{\varepsilon}{\Rightarrow} p'$ and $p' \, G \, q'$. This shows that G is a loose location bisimulation (with respect to the identity relation on locations), therefore $p \approx_{\ell\ell} q$.

Now we show $r \approx_{\ell\ell} s$, by induction on $|p|$. If $|p| = 0$ then $p \approx_{\ell\ell} nil \approx_{\ell\ell} q$, hence $r \approx_{\ell\ell} s$ since $u :: nil \approx_{\ell\ell} nil$ and $(nil \mid t) \approx_{\ell\ell} t$. Now assume $|p| > 0$, and let p' be such that $p \overset{a}{\underset{v}{\Rrightarrow}} p'$ where $v \in loc(p)^*$. We have $(u :: p \mid r) \overset{a}{\underset{uv}{\Rrightarrow}} (u :: p' \mid r)$, therefore there exist q' and s' such that $(u :: q \mid s) \overset{a}{\underset{uv}{\Rrightarrow}} (u :: q' \mid s')$ and $(u :: p' \mid r) \approx_{\ell\ell} (u :: q' \mid s')$. Since s cannot be responsible for the transition $\overset{a}{\underset{uv}{\rightsquigarrow}}$, it is easy to see that the pair of terms $(u :: p' \mid r)$ and $(u :: q' \mid s')$ satisfies the required hypotheses. Then $p' \approx_{\ell\ell} q'$, and by induction $r \approx_{\ell\ell} s'$. Moreover $|p'| = |q'|$ and $|r| = |s'|$, therefore in the transition $(u :: q \mid s) \overset{a}{\underset{uv}{\Rrightarrow}} (u :: q' \mid s')$ there cannot be any communication between q and s, hence $s \overset{\varepsilon}{\Rightarrow} s'$. A symmetric argument shows that there exists r' such that $r \overset{\varepsilon}{\Rightarrow} r'$ and $r' \approx_{\ell\ell} s$. From $r \overset{\varepsilon}{\Rightarrow} r' \approx_{\ell\ell} s \overset{\varepsilon}{\Rightarrow} s' \approx_{\ell\ell} r$ it is easy to conclude that $r \approx_{\ell\ell} s$. □

Now we show the converse of Proposition 5.2 for processes which are dynamically generated by executing finite and restriction-free CCS processes. More precisely, let $\mathrm{CCS_{rf}}$ be the set of finite restriction-free CCS processes. In Section 3 we defined $\mathbb{N}_{\mathrm{r}}(Ag)$ to be the set of terms built on top of Ag agents using all the static constructs except restriction. We aim at showing that for processes of $\mathbb{N}_r(CCS_{rf})$ the loose location equivalence coincides with location equivalence. Clearly CCS_{rf} is closed by transitions, therefore we can use the results of Section 3.

Using the equations SL1-SL7 given in Figure 3 it is easy to convert a term of $\mathbb{N}_r(CCS_{rf})$ to a *parallel form*, or *parform* in short, that is a term of the form $\prod_{i\in I} u_i :: p_i$ (defined up to the laws SL1 and SL2) where $p_i \in CCS_{rf}$ and $i \neq j \Rightarrow u_i \neq u_j$. More precisely:

Lemma 5.4 *For any $p \in \mathbb{N}_r(CCS_{rf})$ there exists a parform $\hat{p} = \prod_{i\in I} u_i :: p_i$ such that $p \equiv \hat{p}$.*

Proof (sketch). By structural induction: if p is a term of CCS_{rf} then we let $\hat{p} = \varepsilon :: p$, and we use SL3. For $u :: r$ one uses the induction hypothesis, and SL4, SL5. For $(r \mid s)$, one possibly uses SL4 (up to SL1 and SL2). For $r[f]$ one uses SL6 and SL7. □

Lemma 5.5 *Let $p = \prod_{i\in I} u_i :: p_i$ and $q = \prod_{j\in J} v_j :: q_j$ be two parforms. If $p \approx_{\ell\ell} q$ and $p_k \not\approx nil$ then there exists h such that $u_k = v_h$ and $p_k \approx_{\ell\ell} q_h$, and $\prod_{i\neq k} u_i :: p_i \approx_{\ell\ell} \prod_{j\neq h} v_j :: q_j$.*

Proof. First we show that if $p_k \not\approx nil$ then there exists h such that $q_h \not\approx nil$ and $u_k = v_h w$. Since $p_k \not\approx nil$ there exists a and p'_k such that $p_k \overset{a}{\underset{\varepsilon}{\Longrightarrow}} p'_k$ (recall that p_k is a CCS term, hence without locations). Let $r = \prod_{i\neq k} u_i :: p_i$. Then $(u_k :: p_k \mid r) \overset{a}{\underset{u_k}{\Longrightarrow}} (u_k :: p'_k \mid r)$, therefore there exists q' such that $q \overset{a}{\underset{u_k}{\Longrightarrow}} q'$ and $q' \approx_{\ell\ell} (u_k :: p'_k \mid r)$. Clearly this implies that there exists $h \in J$ such that $u_k = v_h w$, and q_h is responsible for the action a, that is $q_h \not\approx nil$.

Now we prove the lemma by induction on the cardinality of $I_k = \{\, i \mid p_i \not\approx nil \;\&\; u_k = u_i v \,\}$. If $I_k = \{k\}$ then, by the previous point, there exists h such that $q_h \not\approx nil$ and $u_k = v_h w$. By symmetry, there exists i such that $p_i \not\approx nil$ and $v_h = u_i w'$. Then $u_k = u_i w' w$, therefore $w'w = \varepsilon$, hence $v_h = u_k$. Let $r = \prod_{i\neq k} u_i :: p_i$ and $s = \prod_{j\neq h} v_j :: q_j$. It is easy to see that the terms $(u_k :: p_k \mid r)$ and $(v_h :: q_h \mid s)$ satisfy the hypotheses of the decomposition lemma above; in particular, (i) is satisfied since p_k and q_h are CCS terms, and (ii) is satisfied, because of the minimality of u_k. Then $p_k \approx_{\ell\ell} q_h$, and $\prod_{i\neq k} u_i :: p_i \approx_{\ell\ell} \prod_{j\neq h} v_j :: q_j$.

If $\{k\} \subset I_k$ then there exists $n \in I_k$ such that $u_k = u_n v$ with $v \neq \varepsilon$, that is $I_n \subset I_k$. Then by induction on the cardinality of I_n there exists m such that $v_m = u_n$, $q_m \approx_{\ell\ell} p_n$ and $\prod_{i\neq n} u_i :: p_i \approx_{\ell\ell} \prod_{j\neq m} v_j :: q_j$, and we use again the induction hypothesis regarding $I_k - \{n\}$ to conclude. □

An obvious consequence of this lemma is the following characterization of loose location equivalence on parforms:

Corollary 5.6 *Let $p = \prod_{i\in I} u_i :: p_i$ and $q = \prod_{j\in J} v_j :: q_j$ be two parforms. Then*

$$p \approx_{\ell\ell} q \;\Leftrightarrow\; \{\, u_i \mid p_i \not\approx nil \,\} = \{\, v_j \mid q_j \not\approx nil \,\} \text{ and } u_i = v_j \;\Rightarrow\; p_i \approx_{\ell\ell} q_j$$

Now we can prove that for finite restriction free processes the two location equivalences coincide:

Theorem 5.7 *For any $p, q \in \mathbb{N}_r(\mathrm{CCS_{rf}})$ $p \approx_\ell q \;\Leftrightarrow\; p \approx_{\ell\ell} q$.*

Proof. The "$\Leftarrow$" direction is given by Proposition 5.2. To establish the converse, we show that $\approx_{\ell\ell}$ is a location bisimulation. Let $p \approx_{\ell\ell} q$. Since the transitions $t \overset{\varepsilon}{\Rightarrow} t'$ are the same in the two semantics, we just have to examine the case where $p \xrightarrow[ul]{a} p'$. Let $\hat{p} = \prod_{i\in I} u_i :: p_i$ and $\hat{q} = \prod_{j\in J} v_j :: q_j$ be two parforms such that $p \equiv \hat{p}$ and $q \equiv \hat{q}$. By the Lemma 3.3 there exists $\bar{p}$ such that $\hat{p} \xrightarrow[ul]{a} \bar{p}$ and $p' \equiv \bar{p}$. This means that $\exists k$ s.t. $u = u_k$ and $p_k \xrightarrow[l]{a} p'_k$ with $\bar{p} = (u_k :: p'_k \mid \prod_{i\neq k} u_i :: p_i)$ (up to SL1-SL2). Since $p_k \not\approx nil$, by the Lemma 5.5 above there exists h such that $u_k = v_h$, $p_k \approx_{\ell\ell} q_h$ and $\prod_{i\neq k} u_i :: p_i \approx_{\ell\ell} \prod_{j\neq h} v_j :: q_j$. Then there exists q'_h such that $q_h \overset{a}{\underset{l}{\Rightarrow\!\!\Rightarrow}} q'_h$ and $q'_h \approx_{\ell\ell} p'_k$, but since q_h is a CCS term this means $q_h \overset{a}{\underset{l}{\Rightarrow}} q'_h$. Therefore $\hat{q} \overset{a}{\underset{ul}{\Rightarrow}} \bar{q}$, where $\bar{q} = (v_h :: q'_h \mid \prod_{j\neq h} v_j :: q_j)$ (up to SL1-SL2), hence $\bar{q} \approx_{\ell\ell} \bar{p}$. By the Lemma 3.3 there exists q' such that $q \overset{a}{\underset{ul}{\Rightarrow}} q'$ and $q' \equiv \bar{q}$, hence $q' \approx_{\ell\ell} p'$ (by the Lemma 3.3 and Proposition 5.2). □

In [BCHK91] we said that "introducing locations adds discriminations between processes only as far as their distributed aspect is concerned". This is still true here: let $\mathrm{CCS_{seq}}$ be the set of sequential processes of CCS, that is processes built without the parallel operator. We show that on finite sequential processes, all location bisimulations induced by a reflexive relation R collapse to weak bisimulation:

Corollary 5.8 *For any reflexive relation R on locations, and any finite processes $p, q \in \mathrm{CCS_{seq}}$:*

$$p\,\mathcal{B}(R)\,q \;\Leftrightarrow\; p \approx q$$

Proof. The "$\Rightarrow$" direction is Corollary 3.6. For the converse, due to the Property 2.4, it is enough to show that $p \approx q \;\Rightarrow\; p \approx_\ell q$. It should be clear that for any finite sequential process

r there exists a finite sequential process s written without restriction (i.e. $s \in \mathrm{CCS}_{\mathrm{rf}}$) such that $r \approx_\ell s$. Then we may assume that p and q do not contain the restriction operator, and the statement reduces to $p \approx q \;\Rightarrow\; p \approx_{\ell\ell} q$, which was proved in [BCHK91] (Proposition 3.11). □

Although this result may be derived straightforwardly for finite processes using Theorem 5.7, it does not depend on the hypothesis that p, q are finite. In fact, an analogous result could be obtained for arbitrary processes of $\mathrm{CCS}_{\mathrm{seq}}$ by adapting the proof of [BCHK91] (Proposition 3.11).

We now turn to the relationship between $\approx_\ell$ and distributed bisimulations. Distributed bisimulations are the first attempt of a semantics for CCS taking the distributed nature of processes into account. They have been introduced in [CH89] and [Cas88] and further studied in [Kie89]. We shall not elaborate on this notion here, and refer the reader to the above mentioned works for further information. In [BCHK91] we have shown that the *distributed bisimulation equivalence* $\approx_d$ coincides with the (loose) location equivalence $\approx_{\ell\ell}$ for finite CCS terms written without relabelling and restriction (in fact one could allow relabelling without affecting this result). As a corollary, the three distribution based equivalences $\approx_\ell$, $\approx_{\ell\ell}$ and $\approx_d$ are the same on this language. Another corollary is that we gain another axiomatization for location equivalence on this language, see [Kie89,BCHK91].

In [BCHK91] it was also shown that distributed bisimulation equivalence differs in general from loose location equivalence. The same argument can be used to distinguish distributed bisimulation equivalence from location equivalence: the following two processes are distributed bisimulation equivalent but neither location equivalent nor loose location equivalent.

$$p = (e.(c.\textstyle\sum_a + d.\prod_a + \bar{c}.\sum_b + \bar{d}.\prod_b) \mid (\bar{d}.\sum_b + \bar{c}.\prod_b + c.\sum_a + d.\prod_a))\backslash\{c,d\},$$
$$q = (e.(c.\textstyle\sum_a + d.\prod_a + \bar{c}.\sum_b + \bar{d}.\prod_b) \mid (\bar{c}.\sum_b + \bar{d}.\prod_b + d.\sum_a + c.\prod_a))\backslash\{c,d\}$$

where

$$\begin{aligned} \textstyle\sum_a &= a_1.a_2.nil + a_2.a_1.nil \quad \text{and} \\ \textstyle\prod_a &= a_1.nil \mid a_2.nil \end{aligned}$$

Thus $\approx_d \not\subseteq \approx_\ell$ and $\approx_d \not\subseteq \approx_{\ell\ell}$. Moreover, the example we gave to distinguish loose location equivalence from location equivalence also shows that $\approx_{\ell\ell} \not\subseteq \approx_d$. On the other hand it may be shown that $\approx_\ell \subseteq \approx_d$. Hence we have the following picture for the relationships of the three distribution based equivalences: for finite and restriction-free processes they all coincide; on the whole language CCS, distributed bisimulation equivalence and loose location equivalence are incomparable, while location equivalence is finer than both of them.

6 The Location Preorder

In this section we instantiate the general definition of $\mathcal{B}(R)$ to obtain a preorder on processes which takes into account their degree of parallelism. Roughly speaking, we seek a relation R such that if $p\ \mathcal{B}(R)\ q$ then p and q have similar behaviour but p is possibly more sequential or less distributed than q. For example, let p, q be the processes:

$$p = a.a.a.nil \qquad\qquad q = a.a.nil \mid a.nil$$

Intuitively, p is a sequential shuffle of q. Let us try to relate the behaviours of these two processes using our location transition semantics. Since we look for a *sensible* relation R between the locations of p and q, of the form $R = \{(ul, vl) \mid u\widehat{R}v,\ l \in Loc\}$, we will use at each step the same label to mark corresponding actions of p and q, and try to figure out the relation $\widehat{R}$.

Suppose that q performs the move $\xrightarrow[l_1]{a}$ from its first component, followed by $\xrightarrow[l_2]{a}$ from its second component, to arrive at the state $q' = l_1 :: a.nil \mid l_2 :: nil$. The only way for p to mimic this behaviour is to perform the two actions in sequence, thus becoming $p' = l_1 :: l_2 :: a.nil$. Now the remaining moves of p' and q' are respectively $l_1 :: l_2 :: a.nil \xrightarrow[l_1 l_2 k]{a} l_1 :: l_2 :: k :: nil$ and $l_1 :: a.nil \mid l_2 :: nil \xrightarrow[l_1 k]{a} l_1 :: k :: nil \mid l_2 :: nil$ for some k. Thus the relation $\widehat{R}$ needs to be such that $l_1 l_2\ \widehat{R}\ l_1$, and therefore the inverse of the *suffix* relation is not appropriate.

Conversely, assume that q performs the move $\xrightarrow[l_1]{a}$ from its second component. Then after two steps of execution q comes to the state $q'' = l_2 :: a.nil \mid l_1 :: nil$. The corresponding state of p is again $p' = l_1 :: l_2 :: a.nil$, and the only remaining transitions of p' and q'' are $l_1 :: l_2 :: a.nil \xrightarrow[l_1 l_2 k]{a} l_1 :: l_2 :: k :: nil$ and $l_2 :: a.nil \mid l_2 :: nil \xrightarrow[l_2 k]{a} l_1 :: k :: nil \mid l_2 :: nil$. Hence $\widehat{R}$ must satisfy $l_1 l_2\ \widehat{R}\ l_2$, and this excludes the inverse of the *prefix* relation.

The above example rules out both (the inverses of) the suffix and the prefix relation as candidates for $\widehat{R}$; we have already seen in Section 4 that these relations are not adequate technically, in that they are not preserved by concatenation. On the other hand, the *superword* relation on locations could be an appropriate relation to choose as $\widehat{R}$. Intuitively, if p is a sequentialized version of q, this means that some component of p corresponds to a group of components of q; then, provided at each step the same locations are introduced for corresponding actions, the location u of a component of p will always be a *shuffle* of the locations of the corresponding components of q (and thus a superword of the location of each individual component).

Let $\gg$ denote the *superword* relation on Loc^*. This is the inverse of the *subword* relation, which we note $\ll$. Recall that v is a subword of u, written $v \ll u$, if $v = v_1 \ldots v_k$ and $u = w_1 v_1 \ldots w_k v_k w_{k+1}$, for some collection of words v_i, w_j. Now it is easy to check that the relation R generated by $\widehat{R} = \gg$, which we denote $\gg_\ell$, is a sensible relation on locations, and therefore is a suitable candidate for our theory.

Property 6.1 *The relation* $\gg_\ell \;=\; \{\, (ul, vl) \mid u \gg v,\ l \in Loc \,\}$ *is sensible in the sense of Definition 4.1.*

Since $\mathcal{B}(\gg_\ell)$ is a preorder, we will call it the *location preorder* and denote it by $\sqsubseteq_\ell$. By virtue of the results of the previous section we have a complete axiomatisation of the location precongruence $\sqsubseteq^c_\ell$ over finite terms. The axiom (or more accurately the axiom schema) which is specific to $\sqsubseteq^c_\ell$ is:

$$(\mathrm{GEN}_{\gg_\ell}) \qquad \text{If } \; u \gg v \; \text{ then: } \; <a \text{ at } ux>.\, p \;\sqsubseteq\; <a \text{ at } vx>.\, p$$

Let us examine some more examples.

Example 6.2 For any processes p, q

$$a.(p \mid b.q) + b.(a.p \mid q) \sqsubseteq_\ell a.p \mid b.q$$

Example 6.3

$$rec\,x.\ a.P \sqsubseteq_\ell rec\,x.\ a.P \mid rec\,x.\ a.P$$

To establish this it is sufficient to note that the relation G, consisting of all pairs

$$(u :: rec\,x.\ a.P,\ \ (u_1 :: rec\,x.\ a.P \mid u_2 :: rec\,x.\ a.P))$$

where the word u is a shuffle of the words u_1 and u_2, satisfies $G \subseteq C_{\gg_\ell}(G)$.

Example 6.4 If α is different from a and does not appear in p, q then

$$a.(p \mid q) \sqsubseteq_\ell (a.\alpha.p \mid \overline{\alpha}.q)\backslash\alpha$$

Note that, the process $(a.\alpha.p \mid \overline{\alpha}.q)\backslash\alpha$ could also be expressed as $a.p \not\mid q$, where $\not\mid$ is the leftmerge operator used in papers such as [BK85], [Hen88], [CH89]. So if we were to have $\not\mid$ in our language, the semantic preorder $\sqsubseteq_\ell$ would satisfy

$$a.(x \mid y) \sqsubseteq_\ell a.x \not\mid y$$

However, while in the interleaving theory presented in [BK85] one has the semantic equality $a.x \not\mid y \;=\; a.(x \mid y)$, in our case we would have:

$$a.x \not\mid y \not\sqsubseteq_\ell a.(x \mid y)$$

because $\not\mid$ is a left-parallel operator, and thus in our semantics it would give precedence to the left component but assign independent locations to its two components.

Example 6.5 We have seen in Section 5 that there are two terms, namely $(a.\alpha \mid \bar{\alpha}.b)\backslash\alpha$ and $(a.(\alpha+b) \mid \bar{\alpha}.b)\backslash\alpha$, that are loosely location equivalent but not location equivalent. These two processes are related by the location preorder as follows:

$$(a.(\alpha+b) \mid \bar{\alpha}.b)\backslash\alpha \sqsubseteq_{\ell} (a.\alpha \mid \bar{\alpha}.b)\backslash\alpha \quad \text{and} \quad (a.\alpha \mid \bar{\alpha}.b)\backslash\alpha \not\sqsubseteq_{\ell} (a.(\alpha+b) \mid \bar{\alpha}.b)\backslash\alpha$$

Example 6.6 Let us see why using $\gg_\ell$ instead of $\gg$ is important. This will explain why in the definition of sensible relations R we require that if $u R v$ then u and v end with the same location name (this fact was also used in the Completeness Theorem of Section 4). Let $p = a.((b.\alpha \mid \bar{\alpha}.c)\backslash\alpha + b.c)$ and $q = (a.\alpha \mid \bar{\alpha}.b.c)\backslash\alpha$. These processes

could also be written as $p = a.((b \not| c) + b.c)$ and $q = (a \not| b.c)$. Then $p \not\sqsubseteq_{\ell} q$ since the only possibility for q to match, up to $\gg_\ell$, the sequence of moves $p \overset{a}{\underset{l}{\Longrightarrow}} \overset{b}{\underset{lk}{\Longrightarrow}} l :: (k :: nil \mid c)\backslash\alpha \approx_\ell l :: c$ is $q \overset{a}{\underset{l}{\Longrightarrow}} \overset{b}{\underset{k}{\Longrightarrow}} (l :: nil \mid k :: c)\backslash\alpha \approx_\ell k :: c$, and clearly $l :: c \not\approx_\ell k :: c$ if $l \neq k$. On the other hand, q can match the moves of p up to $\gg$, since $q \overset{a}{\underset{l}{\Longrightarrow}} \overset{b}{\underset{l}{\Longrightarrow}} (l :: nil \mid l :: c)\backslash\alpha$. Indeed we have $p\mathcal{B}(\gg)\, q$, but intuitively we do not want to regard p as less parallel than q since in p the action c is not necessarily spatially dependent on b.

Let us now reconsider the protocol example of the previous section. We saw that $Spec \not\approx_\ell Sys$. On the other hand it is easy to check that the specification is more sequential than the system, namely that $Spec \sqsubseteq_{\ell} Sys$. This is done by showing that the relation:

$$\begin{array}{rcll} S & = & \{ & (u :: Spec,\ (v :: Sender \mid w :: Receiver)\backslash\alpha,\beta\ , \\ & & & (u :: out.\,Spec,\ (v :: \bar{\alpha}.\beta.\,Sender \mid w :: Receiver)\backslash\alpha,\beta\ , \\ & & & (u :: out.\,Spec,\ (v :: \beta.\,Sender \mid w :: out.\bar{\beta}.\,Receiver)\backslash\alpha,\beta\ , \\ & & & (u :: Spec,\ (v :: \beta.\,Sender \mid w :: \bar{\beta}.\,Receiver)\backslash\alpha,\beta\ , \quad \mid \quad u \text{ is a shuffle of } v \text{ and } w\ \} \end{array}$$

is a $\gg_\ell$ – location bisimulation.

To our knowledge there are not many notions of preorder expressing that one process is more sequential than another. An earlier definition of such a preorder was proposed by L. Aceto in [Ace89] for a subset of CCS. This preorder is based on a pomset transition semantics for the language: essentially one process is more sequential than another, in notation $p \sqsubseteq q$, when the pomsets labelling the transitions of p are more sequential than those labelling the transitions of q (this "more sequential than" ordering on pomsets was introduced by Grabowski and Gischer). Thus the intuition underlying Aceto's preorder is somewhat different from ours, in the same way as the causal equivalences mentioned in Section 5, aimed at reflecting causality, are different from the location equivalence $\approx_\ell$, which is designed to reflect distribution in space. Indeed we saw in the previous section that equivalences based on causality and equivalences based on distribution are incomparable in general.

For the preorder we have a similar situation: in fact, the causality-based preorder $\sqsubseteq\!\!\!\!\sim$ and the distribution-based preorder $\sqsubseteq\!\!\!\!\sim_\ell$, turn out to be different even on the small sublanguage without communication and restriction. The following is an example, suggested by L. Aceto, showing that $\sqsubseteq\!\!\!\!\sim \not\subseteq \sqsubseteq\!\!\!\!\sim_\ell$. Let:

$$p = a.b.c + c.a.b + (a \mid b \mid c)$$
$$q = p + a.b \mid c$$

Then $p \sqsubseteq\!\!\!\!\sim q$ but $p \not\sqsubseteq\!\!\!\!\sim_\ell q$ (while we have both $q \sqsubseteq\!\!\!\!\sim p$ and $q \sqsubseteq\!\!\!\!\sim_\ell p$). To see why $p \not\sqsubseteq\!\!\!\!\sim_\ell q$ consider the move $q \xrightarrow[l]{a} l :: b \mid c$, due to the summand $a.b \mid c$ of q. Now p has two ways of doing an a-transition, namely $p \xrightarrow[l]{a} l :: b.c$ and $p \xrightarrow[l]{a} l :: nil \mid b \mid c$, but neither of them is appropriate. It would still be plausible that on the sublanguage, $p \sqsubseteq\!\!\!\!\sim_\ell q$ implies $p \sqsubseteq\!\!\!\!\sim q$.

Having introduced the preorder $\sqsubseteq\!\!\!\!\sim_\ell$, it is natural to consider the associated equivalence, i.e. the kernel $\simeq_\ell =_{\text{def}} \sqsubseteq\!\!\!\!\sim_\ell \cap \sqsupseteq\!\!\!\!\sim_\ell$. All the examples given for $\approx_\ell$ also hold for $\simeq_\ell$. In fact it is clear that $\approx_\ell \subseteq \simeq_\ell$ since we have both $\approx_\ell \subseteq \sqsubseteq\!\!\!\!\sim_\ell$ and $\approx_\ell \subseteq \sqsupseteq\!\!\!\!\sim_\ell$. On the other hand, as could be expected, the kernel of the preorder is weaker than location equivalence, that is $\simeq_\ell \not\subseteq \approx_\ell$. An example is:

$$a.a.a + (a \mid a \mid a) \qquad \text{and} \qquad a.a.a + a.a \mid a + (a \mid a \mid a)$$

These two processes are obviously not equivalent w.r.t. $\approx_\ell$, but they are equivalent w.r.t. $\simeq_\ell$ because $a.a.a \sqsubseteq\!\!\!\!\sim_\ell a.a \mid a \sqsubseteq\!\!\!\!\sim_\ell a \mid a \mid a$.

We have seen in the previous section that on *sequential* CCS processes all the *plb*'s based on sensible relations collapse to weak bisimulation. For the preorder $\sqsubseteq\!\!\!\!\sim_\ell$ we shall prove a stronger result, similar to that given in [Ace89] for the causal preorder $\sqsubseteq\!\!\!\!\sim$, namely that for $p \sqsubseteq\!\!\!\!\sim_\ell q$ to imply $p \approx q$ it is enough that the first process p be sequential.

Recall that CCS_{seq} is the set of sequential processes of CCS, that is processes built with all the constructs of CCS except the parallel operator, and that $\equiv$ is the congruence over $\mathbb{P}$ generated by the equations SL1-SL8 given in Figure 3, Section 3. The following lemma establishes that processes derived from CCS_{seq} are always of the form $u :: p$, where $p \in \text{CCS}_{\text{seq}}$, up to the congruence $\equiv$ (more precisely up to the laws SL5, SL7, SL8).

Lemma 6.7 *Let* $p \in \mathrm{CCS_{seq}}$ *and* $u \in Loc^*$. *Then:*

(1) *If* $u :: p \xrightarrow[ul]{a} p'$ *then*
$\exists\, r \in \mathrm{CCS_{seq}}$ *s.t.* $p' \equiv ul :: r$.

(2) *If* $u :: p \xrightarrow{\mu} p'$ *then* $\exists r \in \mathrm{CCS_{seq}}$ *s.t.* $p' = u :: r$.

Proof. (1) The transition $u :: p \xrightarrow[ul]{a} p'$ is inferred from a transition $p \xrightarrow[l]{a} p''$ such that $p' = u :: p''$. Thus all we have to show is that there exists $r \in \mathrm{CCS_{seq}}$ such that $p'' \equiv l :: r$, since then we will have $p' \equiv u :: l :: r \equiv ul :: r$ by SL5. To show that $\exists\, r \in \mathrm{CCS_{seq}}$ s.t. $p'' \equiv l :: r$, we use induction on the proof of $p \xrightarrow[l]{a} p''$. Note that all the transition rules in Figure 2 preserve the form of the derivative except for LT1, LT6, LT7. Thus we only have to consider these three cases, since for the others we have the result immediately by induction.
- Suppose the last rule applied is LT1. Then p is of the form $a.q$ and $p'' = l :: q$. Moreover $q \in \mathrm{CCS_{seq}}$ because $p \in \mathrm{CCS_{seq}}$.
- Suppose now the last rule applied is LT7. Here $p = q[f]$, and $q[f] \xrightarrow[l]{f(a)} q'[f]$ is inferred from $q \xrightarrow[l]{a} q'$. By induction $\exists\, s \in \mathrm{CCS_{seq}}$ s.t. $q' \equiv l :: s$. Then $q'[f] \equiv (l :: s)[f] \equiv l :: (s[f])$ by SL7. The case of LT6 is treated similarly, using law SL8.
(2) This is trivial since $p \xrightarrow{\mu} p'$ can only be proved using ST3, and obviously $\mathrm{CCS_{seq}}$ is closed w.r.t the transitions $\xrightarrow{\mu}$. $\square$

Recall that $\mathbb{N}(\mathrm{CCS})$ was defined in Section 3 to be the set of processes built on top of CCS processes using the static constructs. Let $Oloc(p)$ be the set of (immediately) *observable locations* of p, defined as follows:

$$Oloc(p) =_{\mathrm{def}} \{ u \in Loc^* \mid \exists p'.\, p \overset{a}{\underset{ul}{\Longrightarrow}} p' \}$$

Note that for any process $p \in \mathbb{P}$ we have $Oloc(p) \subseteq loc(p)^*$. For example, if $p \in \mathrm{CCS}$ we have $Oloc(p) \subseteq \{\varepsilon\}$, while for $p \in \mathbb{N}(\mathrm{CCS})$ we have the following property:

Fact 6.8 *If* $p \in \mathbb{N}(\mathrm{CCS})$ *then* $p \overset{a}{\underset{ul}{\Longrightarrow}} p' \;\Rightarrow\; Oloc(p') \subseteq Oloc(p) \cup \{ul\}$

For $X \subseteq Loc^*$, we will use the notation $u \gg X$ to mean: $\forall v \in X.\ u \gg v$. Note that, as a consequence of the Lemma 3.3, we have $\equiv \subseteq \underset{\sim}{\sqsubset}_{\ell}$. We may now prove our result.

Proposition 6.9 *If* $p \in \mathrm{CCS_{seq}}$ *and* $q \in \mathrm{CCS}$, *then* $p \underset{\approx}{\sqsubset}_{\ell} q \;\Leftrightarrow\; p \approx q$.

Proof. The implication $p \underset{\approx}{\sqsubset}_{\ell} q \Rightarrow p \approx q$ is true in general. We show that if p is sequential we also have $p \approx q \Rightarrow p \underset{\approx}{\sqsubset}_{\ell} q$. To this purpose we use the fact that $\approx = \mathcal{B}(U_\ell)$ (*cf* Lemma

4.2). We prove that the relation:

$$S = \{ (u :: p, q) \mid u :: p\ \mathcal{B}(U_\ell)\ q,\ p \in \mathrm{CCS_{seq}}\ , q \in \mathbb{N}(\mathrm{CCS}) \text{ and } u \gg Oloc(q) \}$$

is a $\gg_\ell$ – location bisimulation up to the equivalence $\equiv$. More precisely we show that if $u :: p \overset{a}{\underset{ul}{\Longrightarrow}} p'$ then $\exists v, q', p''$ s.t. $q \overset{a}{\underset{vl}{\Longrightarrow}} q'$, with $u \gg v$ and $p' \equiv p''\, S\, q'$ (and similarly for ε-moves). Suppose $u :: p \overset{a}{\underset{ul}{\Longrightarrow}} p'$. Then by Lemma 6.7 $\exists\, r \in \mathrm{CCS_{seq}}$ such that $p' \equiv ul :: r$. Since $u :: p\, \mathcal{B}(U_\ell)\, q$, there must exist v, q' such that $q \overset{a}{\underset{vl}{\Longrightarrow}} q'$, with $p'\, \mathcal{B}(U_\ell)\, q'$, and $u \gg v$ because $u \gg Oloc(q)$. Then also $ul :: r\, \mathcal{B}(U_\ell)\, q'$, since $\equiv \subseteq \mathcal{B}(U_\ell)$. Now $Oloc(q') \subseteq Oloc(q) \cup \{vl\}$ by Fact 6.8 above, and obviously $ul \gg Oloc(q)$ (since $u \gg Oloc(q)$) and $ul \gg vl$, thus $ul \gg Oloc(q')$ and therefore $p' \equiv ul :: r\ S\ q'$.

Suppose now $u :: p \overset{\varepsilon}{\Rightarrow} p'$. By Lemma 6.7 $\exists\, r \in \mathrm{CCS_{seq}}$ such that $p' = u :: r$. Then, since $u :: p\ \mathcal{B}(U_\ell)\ q$, we have $q \overset{\varepsilon}{\Rightarrow} q'$ with $p'\ \mathcal{B}(U_\ell)\ q'$, and thus $p'\ S\ q'$.

In particular for $p \in \mathrm{CCS_{seq}}$ and $q \in \mathrm{CCS}$ we have $p \approx q \;\Rightarrow\; \varepsilon :: p\ S\ q$, therefore $p \approx q \;\Rightarrow\; p \sqsubseteq\!\!\!\!\sim_\ell q$. □

In most of the examples considered so far to illustrate the preorder $\sqsubseteq\!\!\!\!\sim_\ell$ – or at least in the "concrete" examples – the process on the left in $p \sqsubseteq\!\!\!\!\sim_\ell q$ was actually a sequential process. As a result of the above proposition, proving $p \sqsubseteq\!\!\!\!\sim_\ell q$ in this case reduces to showing $p \approx q$. For a concrete example where the specification is not a sequential process we refer the reader to [Ace89].

References

[Ace89] L. Aceto. On relating concurrency and nondeterminism. Report 6/89, Computer Science, University of Sussex, Brighton, 1989.

[Ace91] L. Aceto. A static view of localities. Report 1483, INRIA, 1991.

[BCHK91] G. Boudol, I. Castellani, M. Hennessy, and A. Kiehn. Observing localities. Report 4/91, Sussex University, and INRIA Res. Rep. 1485, 1991.

[BK85] J. Bergstra and J.W. Klop. Algebra of communicating processes with abstraction. *Theoretical Computer Science*, 37:77–121, 1985.

[Cas88] I. Castellani. *Bisimulations for Concurrency.* Ph.d. thesis, University of Edinburgh, 1988.

[CH89] I. Castellani and M. Hennessy. Distributed bisimulations. *JACM*, 10(4):887–911, 1989.

[DD89] Ph. Darondeau and P. Degano. Causal trees. In *Proc. ICALP 88*, volume 372 of *LNCS*, pages 234–248, 1989.

[DD90] Ph. Darondeau and P. Degano. Causal trees: interleaving + causality. In *Semantics of Systems of Concurrent Processes*, number 469 in LNCS, pages 239–255, 1990.

[DDNM87] P. Degano, R. De Nicola, and U. Montanari. Observational equivalences for concurrency models. pages 105–129, 1987.

[Hen88] M. Hennessy. Axiomatising finite concurrent processes. *SIAM Journal of Computing*, 17(5):997–1017, 1988.

[HM85] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *JACM*, 32(1):137–161, 1985.

[Kie89] A. Kiehn. Distributed bisimulations for finite CCS. Report 7/89, University of Sussex, 1989.

[Kie91] A. Kiehn. Local and global causes. Report, to appear, Technische Universität München, 1991.

[Mil79] R. Milner. Flowgraphs and flow algebra. *JACM*, 26(4):794–818, 1979.

[Mil80] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. 1980.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[MY91] U. Montanari and D. Yankelevich. Partial order localities. draft, Università di Pisa, 1991.

[vGG90] R. van Glabbeek and U. Goltz. Equivalences and refinement. In *Semantics of Systems of Concurrent Processes*, number 469 in LNCS, pages 309–333, 1990.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique.

ISSN 0249-6399